AN ABSTRACT OF THE THESIS OF

Nudtapon Nudtasomboon for the degree of Doctor of Philosophy in Industrial Engineering presented on March 12, 1993.

Title: A Methodology for the Multi-Objective, Resource-Constrained Project Scheduling Problem

Abstract approved: Redacted for Privacy
Sabah U. Randhawa

This study is concerned with the problem of resource-constrained project scheduling which includes splittable and nonsplittable jobs, renewable and nonrenewable resources, variation in resource availability, time-resource tradeoff, time-cost tradeoff, and multiple objectives.

The problem is formulated as a zero-one integer programming model. A specialized solution technique is developed for the preemptive goal programming, resource-constrained project scheduling problem for time, cost, and resource leveling objectives. In addition, single objective algorithms are also provided for the time, cost, and resource leveling objectives. These algorithms are based on the idea of the implicit enumeration process, and use the special structures of the problem to expedite the search process.

Computer-generated problems are used to test each of the single objective algorithms. The results show that the algorithms give optimal solutions to tested problems with time and cost objectives using a reasonable computation time; however, heuristic solutions are more feasible for problems

with resource leveling objective. The multiple objective algorithm is illustrated through application to a warehouse project problem.

A Methodology for the Multi-Objective, Resource-Constrained Project Scheduling Problem

by

Nudtapon Nudtasomboon

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Doctor of Philosophy

Completed March 12, 1993

Commencement June 1993

APPROVED:

Redacted for Privacy

Associate Professor of Industrial & Manufacturing Engineering in charge of major

Redacted for Privacy

Head of Department of Industrial & Manufacturing Engineering

Redacted for Privacy

Dean of Graduate School

Date thesis is presented March 12, 1993

Typed by Nudtapon Nudtasomboon for Nudtapon Nudtasomboon

ACKNOWLEDGEMENTS

I am deeply grateful to my major professor, Dr. Sabah Randhawa, for his invaluable advice, constant support, encouragement, and friendship throughout my doctoral studies.

I would like to thank Dr. Harold Pritchett and Dr. Eldon Olsen for his friendship and helpful advice during my graduate studies. I am grateful to Dr. David Birkes for his help and kind assistance. I would also like to thank Dr. Paul Paschke for his suggestion as well as for serving on my committee.

Thanks to Oregon State University Department of Industrial and Manufacturing Engineering for the financial support during my doctoral studies.

Finally, I would like to thank my best parents for everything throughout my life and to my special friend, Sauwaluck Pakotiprapha, for her encouragement and patience during my graduate studies.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

A METHODOLOGY FOR THE MULTI-OBJECTIVE, RESOURCE-CONSTRAINED PROJECT SCHEDULING PROBLEM

CHAPTER 1

INTRODUCTION

1.1 Statement of the Problem

Scheduling is one of the three basic managerial functions (planning, scheduling, and control) that determines the success of any project. Project schedules, determined using techniques such as CPM and PERT, may not be applicable in practice if resource constraints are not included. The resource-constrained project scheduling problem is concerned with the allocation of limited resources to competing activities with the objective of optimizing some specified performance measures. Past research treats the resource-constrained project scheduling problem as a single objective problem where a minimum project completion time is desired under limited resources. It does not account for variations in resource levels. Fluctuations in resource levels can generate higher costs and reduce efficiency of the resources.

Leveling resources in the resource-constrained project scheduling problem may not necessarily result in reduction of project costs. This is reasonable because traditional network techniques assume that estimates of activity durations are made independently of each other without considering factors such as different activities competing for the same resource. However, if more than one mode of operation are allowed for each activity, where each mode reflects a different combination of resource consumption and different activity durations, a higher efficiency can be expected from the use of resource leveling.

The use of multiple performing modes also incorporates time-cost tradeoff concept since each mode consists of a different combination of resources resulting in different activity direct costs. The traditional resource-constrained project scheduling problem assumes the availability of resources to be limited at each time period. However, there are some resources whose availability is constrained over the project life rather than at each time period. The traditional approach also assumes that an activity cannot be interrupted once it starts, even though the preemption of some activities can increase the efficiency of the schedule. In addition, a more effective resource schedule which will reduce the overall project total cost can be obtained if the resource-constrained project scheduling problem is treated as a multiple objective problem rather than a single objective problem as before.

1.2 Objective and Scope of the Study

The objective of this research is two-fold:

1. To develop a multiple objective project scheduling model under resource constraints. Objectives include time, cost, and resource balancing.
2. To develop a solution approach for the multiple

objective problem formulated in Step (1).

The model assumes that activity time estimates are a function of resources which can be represented by a different performing mode, and the estimates of individual activity durations depend on the number of resources allocated to competing activities. Both splittable (interruptable) jobs and nonsplittable (non-interruptable) jobs are allowed in the model. Three types of resources are considered: those constrained at each time period (renewable), those constrained over project life (nonrenewable), and those that are constrained both at each time period and over project life (doubly-constrained). A balance among multiple objectives is desired under resource constraints. System objectives include: minimum project completion time, minimum project total cost, and minimum variation in resource levels.

1.3 General Approach

A zero-one integer programming model is developed to represent the resource-constrained project scheduling problem. In addition to the technological constraints, objective functions are developed for minimizing total project throughput time, minimizing makespan, minimizing total lateness or lateness penalty, minimizing project overhead cost (indirect cost), minimizing project direct cost, minimizing project total cost, and resource leveling objectives. However, only the objectives of minimizing total project throughput time, minimizing project total cost, and resource leveling are the focus in this research since they are of prime interest to many construction projects.

Due to the computational inefficiency of large scale integer optimization models, a specialized solution technique is developed. The technique is based on the implicit enumeration procedure developed by Talbot (1978, 1982). Talbot's algorithm is a two-stage approach which includes the labeling process and the enumeration process. The labeling process is performed at the first stage to specify the order of jobs, modes, and resources. The enumeration process attempts to systematically evaluate all possible solution candidates. During the process, an augmentation process attempts to assign a new feasible job to a partial schedule, and the backtracking process attempts to reschedule a job previously assigned to the partial schedule. Optimality is reached when all jobs are systematically evaluated.

The Talbot's algorithm is improved and extended to include the following features:

(1) Skip mode process which is used to eliminate certain modes of a job from consideration.

(2) Bound checking process which is designed to handle nonrenewable resource conflicts more effectively.

(3) Flexible backtracking process which allows jobs to be rescheduled according to a selected backtracking rule.

(4) A procedure for job splitting is provided, thus, the algorithm can handle both splittable jobs and nonsplittable jobs.

(5) An algorithm for the resource leveling problem is developed.

(6) The algorithm can be modified to handle the multi-project scheduling problem.

(7) An algorithm for the multiple objective goal programming problem is also developed.

Algorithms for a single objective resource-constrained project scheduling problem are provided for each of the three primary objectives: time minimization, cost minimization, and resource leveling. A multiple objective algorithm for these three objectives is then developed. Computer implementation is developed for each algorithm using the C programming language.

Computer-generated problems are used to test each of the single objective algorithms. The experiments are mainly focused on testing the computational efficiency of the backtracking rules, and comparing them with Talbot's algorithm, where applicable. The results show that there are no differences among backtracking rules in the time minimization algorithm, but all backtracking rules require significantly less computation time than the algorithm developed by Talbot. In the cost minimization algorithm, some backtracking rules require less computation time than the others, and all backtracking rules require significantly less computation time than the algorithm developed by Talbot. Since the computation time required to reach optimality of the resource leveling problem is significant, an upper bound set for the computation time is defined for the tested problems. The heuristic solutions obtained are used for comparing between backtracking rules.

The best backtracking rules for each objective are then selected based on the experimental results, and are used in the multiple objective algorithm. The multiple objective algorithm is implemented for a warehouse project problem. Three goals are considered: (1) the highest priority goal is to minimize the lateness beyond a desired due date, (2) the second priority goal is to minimize the total project cost under the desired due date or the objective value obtained from the first priority goal, and (3) the third priority goal is to minimize the resource deviations from a desired level without violating the objective values obtained from the higher priority goals.

The warehouse project consists of twelve activities, including one dummy terminal activity. Each activity can be performed by using one of three available modes of operation, and activity durations range from 22 to 138 time units. Activities compete for one renewable resource type and one nonrenewable resource type. Several resource combinations are tested, including both loose and tight resource limits. Optimality is reached for both time and cost objectives within one second of computation time (on a machine with 33MHz, 486DX processor) in the problem with loose resource limit, and heuristic solutions are obtained for the resource leveling objective. Heuristic solutions are also obtained for the three objectives in the problem with tight resource limit.

1.4 Organization of the Thesis

Chapter 2 describes problem investigated in details, summarizes prior research works, and discusses the limitations of existing approaches. Chapter 3 gives a brief summary of how the solution approach is developed. Chapter 4 explains the mathematical formulation of the problem and the experience with the mathematical models. Chapter 5 describes the Talbot's algorithm, improvements and extensions, and the computerized algorithms and their implementation. Chapter 6 explains how tested problems are generated and used in the experiments, and summarizes the results of the tests performed for the single objective algorithms and of the application of the multiple objective algorithm. Chapter 7 summarizes the works done in this study and their limitations, and gives recommendations for further research.

CHAPTER 2

BACKGROUND

2.1 Problem Description

CPM (Critical Path Method) and PERT (Program Evaluation and Review Technique) have been widely used as management tools to effectively plan and control projects. The basic concept of CPM and PERT is to complete the schedule of a project as soon as possible under specified technological constraints assuming that unlimited amount of resources are available. The assumption of unlimited resources may not be justified in many circumstances since only a fixed amount of resources are available or the cost of acquiring additional resources is very high. In such cases, resource constraints should be considered as a factor that can significantly affect the schedule.

Resource scheduling problems may be divided into two basic categories: resource allocation (fixed resource limits scheduling) and resource leveling (Wiest and Levy 1977; Moder et al. 1983).

1. <u>Resource allocation problems</u> involve an allocation of limited resources to competing activities with the objective of minimizing project duration.
2. <u>Resource leveling problems</u> assume that sufficient resources are available (i.e., unlimited resources) and attempt to reduce resource fluctuations and to

maintain more stable resource levels without stretching a project beyond its optimal duration.

A resource may be classified as one of three categories: renewable, nonrenewable, and doubly-constrained. In resource-constrained (resource allocation) scheduling problems, the availability of renewable resources, such as machine and manpower, is limited at each time period. The availability of nonrenewable resource, such as money, is constrained over the project life. If a resource is constrained both at each time period and over the project life, the resource is said to be doubly-constrained.

In resource scheduling, estimates of activity durations are made independently of each other without considering factors such as different activities competing for the same resource. Furthermore, a normal level of resources (the optimum level of resources under the consideration of expedient completion and minimum costs) is assumed for each activity. In many circumstances, these assumptions are made even though there appears to be some relationship between activity durations and the number of resources applied to them. Such relationships have not been traditionally included in estimating activity durations when resource scheduling is performed.

The interaction between activity durations and the number of resources can be represented by different performing modes of each activity. Each mode reflects a different combination of resources, and has a different activity duration. The time-resource interactions also contain the characteristics of time-cost tradeoffs, which can be represented by a discontinuous cost-time curve as defined by Wiest and Levy (1977). The reason is that each mode of operation consists of a different combination of resources which also reflects a difference in an activity's direct cost.

As mentioned earlier, the objective of resource allocation is to allocate limited resources to competing activities in order to minimize project duration. However, most of resource allocation models do not take into consideration fluctuations in resource levels. Fluctuations in resource levels will increase total project costs due to the costs of hiring, firing, and training human resources, and the costs of leasing or purchasing additional units of equipment. In order to obtain the minimum cost schedule, the resource fluctuations should be minimized or savings due to stable resource levels should be compared to the costs incurred by extending project completion time.

Many important characteristics of project scheduling problem will be considered in this research including activity preemption, renewable and nonrenewable resources, variation in resource availability, time-resource tradeoffs, time-cost tradeoffs, and multiple objectives. Two types of jobs are considered, splittable and nonsplittable jobs. It is assumed that a splittable job can be interrupted (preempted) any number of times, and resume in the same mode after the interruption. A nonsplittable job is a job that must be performed continuously without interruption. At least one mode of operation is available for each job. Each mode reflects a different combination of resources and has a different job duration. Different categories of resources including renewable, nonrenewable, and doubly-constrained are allocated to each job to satisfy multiple conflicting objectives such as project completion time, project total cost, and resource leveling.

2.2 Literature Review

The literature related to resource scheduling problems will be reviewed including both resource-constrained and resource leveling problems. Also, approaches for multiple objective problems will be addressed.

2.2.1 Resource-Constrained Project Scheduling Problem

The resource-constrained project scheduling problem is concerned with the allocation of scare resources to competing activities with the objective of optimizing some specified performance measures such as time minimization.

Several optimization techniques have been developed for solving the resource-constrained project scheduling problem. Optimal solutions of resulting models can be obtained by using mathematical programming approaches or specialized solution techniques.

The zero-one integer linear programming model has been widely used to formulate the resource-constrained project scheduling problem. The first published integer linear programming (ILP) formulation for the resource-constrained project scheduling problem was presented by Wiest (1963). He modified the job-shop formulation developed by Bowman (1959), and used zero-one decision variables to define the status (active or inactive) of jobs at each time period. He concluded that the use of such formulation was computational inefficient even for a small problem. Other early ILP formulations including those by Hadley (1964), and Brand, Meyer and Shaffer (1964) show outcomes similar to the Wiest's result.

A significant improvement in this area resulted from the model developed by Elmaghraby (1969). The model modified the n-job, m-machine scheduling model developed earlier by Elmaghraby (1967), and used zero-one decision variables to define start times for jobs resulting in a significant reduction of the number of variables and constraints. Pritsker, Watters and Wolfe (1969) presented a zero-one formulation for multiproject scheduling under limited resources. Their model has been widely referenced in this area (Hall, 1980). The model uses zero-one decision variables to define completion times of jobs. This formulation results in a relatively small number of variables and constraints. The model could be used to formulate different objective functions including project throughput time, makespan, and lateness penalty under some additional constraints such as substitution of resources, concurrency of jobs, and job splitting.

Since the mathematical formulation approach involves a large number of variables and constraints resulting in computational inefficiencies for the optimization model, special algorithms have been developed for solving the resource-constrained project scheduling problems. These include a bounded enumeration approach, a branch-and-bound (skiptracking) approach, and an implicit enumeration (backtracking) approach.

Davis (1968), and later Davis and Heidorn (1971), presented the bounded enumeration approach for the project scheduling problem under multiple resource constraints based on the technique originally developed for the assembly line balancing problem (Gutjahr 1963, Gutjahr and Nemhauser 1964). In the algorithm, the duration of each job is divided into a series of unit duration tasks, and the original problem is then transformed into an equivalent shortest-route problem. The algorithm can handle job splitting and varying resource usage levels without significantly affecting the computation time. However, the number of feasible precedence subsets can cause the computer memory problem as the problem gets larger.

The branch-and-bound method is a search procedure which involves partitioning a problem into subproblems (branching) and evaluating these subproblems (computing bounds). Johnson

(1967) developed a branch-and-bound approach for the single resource-constrained project scheduling problem. A similar approach for the multiple resource-constrained problem was later developed by Stinson (1976, 1978). In Stinson's procedure, nodes in the branch-and-bound tree represent feasible partial schedules which simultaneously consider both precedence and resource constraints. With the use of "Dominance Pruning" and "Lower Bound Pruning", nonoptimal portions of the tree can be eliminated from consideration increasing the efficiency of the algorithm. The branch-and-bound procedure of Stinson shows a significant improvement in computation time. However, it requires a large amount of computer memory storage even for a small problem.

Patterson and Huber (1974) presented a horizon-varying, zero-one approach to the resource-constrained project scheduling problem. The algorithm uses bounding techniques including a minimum bounding, a maximum bounding, and a binary search to test for the feasibility of a series of zero-one programming problems rather than solving the problem optimally. The approach was reported to be more effective than zero-one programming without bounding, and comparable with other enumerative procedures such as that of Davis (1968, 1971).

A better implicit enumeration (zero-one programming) algorithm was developed by Patterson and Roth (1976). The algorithm exploits the special structure of the zero-one formulation adopted from Elmaghraby's formulation to strengthen the Balas's implicit enumeration algorithm (1965). A similar, but superior, approach was developed by Talbot (1976) and Talbot and Patterson (1978) for solving the resource-constrained project scheduling problem. Talbot's algorithm is also based on the Balas's implicit enumeration algorithm (1965) and the improved implicit enumeration by Geoffrion (1969). The major differences between Talbot's algorithm and the Balasian algorithm are (1) the use of nonnegative integer variables rather than zero-one binary integer variables, and (2) the use of job numbers in the augmentation process rather than feasibility tests. In this algorithm, all possible job completion times are systematically evaluated (enumerated) using the special problem structure. Talbot also introduced a stronger fathoming technique, called a network cut, to eliminate inferior candidate problems from explicit consideration.

Talbot (1982) used the same approach without the network cut concept to solve the nonpreemptive class of the resource-constrained project scheduling problem in which the job duration is a function of the resources consumed. Different types of resources are allowed, including renewable, nonrenewable, and doubly-constrained. The algorithm also extends the earlier algorithm to include the labeling process which is used to define the problem as an integer programming problem based on a heuristic scheduling rule. In addition to the time minimization algorithm, his algorithm can be modified to develop an algorithm for monetary objective functions. Furthermore, his algorithm can be used to bridge the gap between the capital budgeting problem and the time-resource tradeoff problem through the use of nonrenewable resource concept. Since the solution approach developed in this research builds on the Talbot's algorithm, a detailed discussion of Talbot's algorithm is given in Chapter 5.

Past research on optimization procedure does not show much success in its applicability to large-scale problems. This has resulted in development of numerous heuristics for the resource-constrained project scheduling problem. These heuristics produce so-called good feasible solutions, but not optimal solutions. Heuristic solutions can be obtained by allocating scare resources to competing activities according to heuristic rules. Many heuristic procedures are summarized and discussed by Davis (1973). Comparative studies among different heuristics can be found in Patterson (1973), Davis and Patterson (1975), and Kurtulus and Narula (1985).

2.2.2 Resource Leveling Problem

The resource leveling problem attempts to reduce resource fluctuations to maintain more stable resource levels without extending a project completion time beyond its optimal duration assuming that unlimited amount of resources are available.

Very few optimization procedures have been developed to solve the resource leveling problem. Mandeville (1965), also summarized in Moodie and Mandeville (1965), developed a mathematical model for the resource leveling problem by modifying the Bowman's integer linear programming formulation of the assembly line balancing problem. Ahuja (1976) presented an integer linear programming formulation for minimizing resource variations between consecutive periods, and solved it by using his own algorithm which explicitly enumerates all possible combinations of activity float times. Elmaghraby (1977) also presented an integer linear programming model for the preemptive case of the multiple resource leveling problem. He used zero-one decision variables to define the fraction of work accomplished by a job at a certain time, and tried to minimize total cost incurred by resource variations between consecutive periods and a delay beyond due date. In recent research, Easa (1989) developed an integer linear programming model for a single resource leveling problem by using the scheduling results from the critical path method and the resource consumption of activities as input to the model. The model can be used to handle different objective functions including minimization of absolute deviations between the resource requirements and a uniform resource level, between consecutive periods, and between the resource requirements and desirable nonuniform resource levels.

Numerous heuristic procedures have been developed for the resource leveling problem. Examples include Burgess and Killebrew (1962), Galbreath (1965), Shaffer et al. (1965), Antill and Woodhead (1970), Woodworth and Willie (1975), Ahuja (1976), and Harris (1978). The basic concept of these heuristics is to reschedule noncritical activities within the limits of available float according to some heuristic rule to achieve a better distribution of resource usage.

2.2.3 Multiple Objective Procedures

The multiple objective project scheduling problem involves the assignment of the project's activities according to their precedence relationships under some specified constraints such as limited resources to satisfy multiple performance measures.

Several mathematical formulations have been proposed for the multiple objective project scheduling problem. These formulations use the goal programming model to formulate the problem which can be solved by using a technique such as branch-and-bound, implicit enumeration, or cutting plane method. Early applications of multiple objective procedures to the project scheduling problem include the linear goal programming model formulation for the project crashing problem developed by Lee, Moore and Clayton (1976). The model extends the linear programming formulation of the time-cost crashing model to include various managerial objectives. Hanan (1978) demonstrated how objectives other than time and cost can be included in a linear goal programming model. Such objectives include share of the market, completion time of individual jobs, contractual agreements, and scarcity of resources.

An integer (zero-one) goal programming formulation was developed by Wieters (1979) to analyze the multiple project scheduling problem competing for the same resource. The model used the results from the critical path method as input, and the integer goal programming model was formulated to minimize the cost for all projects and the costs associated with early or late completion of individual projects.

Lee and Olson (1984) also introduced zero-one goal programming approach to the application of multi-project scheduling problem based on Bowman's zero-one linear programming formulation. He presented an example model for the zero-one goal programming formulation including several goals such as completion of projects in minimum time, completion of projects on or before penalty times, and completion of projects under resource limitations. The model was solved by using the branch-and-bound goal programming technique; a zero-one linear programming technique was recommended for large models by solving a sequence of linear programming models equivalent to the goal programming formulation (Lee and Olson in Project Management: Methods and Studies, 1985).

A specialized optimization procedure for the multiple objective resource-constrained project scheduling problem was developed by Slowinski (1981). The algorithm assumes that jobs can be preempted any number of times, and each job can have more than one performing mode of operation. Also, different categories of resources are used including renewable, nonrenewable, and doubly-constrained. The algorithm creates all non-empty and different resource feasible subsets which do not violate the resource constraints, then use either one-stage or two-stage multiobjective linear programming (MOLP) methods to solve the problem.

Slowinski (1981) identified the problem of optimally allocating limited resources of different categories among non-splittable jobs with discrete resource requirements as a possible area of further research. A resource-constrained project scheduling with time-resource tradeoff was later developed by Talbot (1982) to deal with a nonpreemptive case where jobs are not allowed to split. However, the algorithm is limited to a single objective such as time or cost minimization.

Recent research by Norbis and Smith (1988) introduced a multiobjective, multi-level heuristic for solving the resource-constrained project scheduling problem in addition to their multiobjective zero-one mathematical formulation. Mohanty and Siddiq (1989) made a comparison between the integer goal programming model and a simulation model based on multiple performance measures such as project slippage, resource-constrained scheduling efficiency, total project delay, weighted total delay, and total resource-idle time. The authors concluded that the integer goal programming model was quite effective for the problems attempted, and the simulation model frequently gave results which are comparable to their corresponding optimum values.

2.3 Limitations of Existing Approaches

1. The focus in prior research has been time minimization in the resource-constrained project scheduling problem. There has been little emphasis on other objectives including cost minimization and resource balancing.
2. The models formulated focus on only selective characteristics. For example, most models ignore time-resource tradeoff, and assume that an activity can be performed by using only one mode of operation.
3. Most models are single-objective models. There has been very limited research on modeling multiple objectives in the area of resource-constrained project scheduling.
4. Since there has been limited effort in developing the multiple objective model, there has been even less of an effort in developing efficient solution algorithms for the multiple objective model. Solution techniques for the integer goal programming formulations are computational

intensive, particularly for microcomputer-based system.

The proposed research will consider project scheduling characteristics not included in prior research works. These characteristics are:

(1) splittable and nonsplittable jobs

(2) renewable and nonrenewable resources

(3) variation in resource availability

(4) time-resource tradeoff

(5) time-cost tradeoff

(6) multiple objectives.

The objectives considered in this research focus in the area of construction industry. Primary objectives in this area are time-related, cost-related, and resource leveling. The resource-constrained project scheduling problem containing the above characteristics will be formulated as an integer goal programming model, and a specialized technique will be developed for its solution.

CHAPTER 3

GENERAL METHODOLOGY

The steps followed in developing the solution approach are summarized below.

1. Development of the mathematical model.
2. Development of the solution approach:
   (a) Modeling basic concepts using Talbot's algorithm.
   (b) Extensions and enhancements of Talbot's algorithm.
3. Development of computerized algorithms for each objective, and for the multiple objective problem.
4. Implementation of the computerized algorithms.
5. Testing single objective algorithms and evaluating heuristic rules using the experimental design approach.
6. Application of the multiple objective algorithm to the warehouse project problem.

Explanation of the above methodology is divided into several chapters to enhance clarification and understanding. Chapter 4 explains the mathematical model (Step 1). Chapter 5 describes the Talbot's algorithm, extensions and enhancements, and the computerized algorithms and their implementation (Steps 2 through 4). Chapter 6 focuses on the tests performed for the single objective algorithms and on the application of the multiple objective algorithm (Steps 5 and 6).

CHAPTER 4

MATHEMATICAL MODEL

4.1 Mathematical Formulation

A zero-one integer programming model was developed to model the resource-constrained project scheduling problem. The model characteristics and assumptions are summarized below.

(1) Two types of jobs are available, splittable and nonsplittable jobs. It is assumed that a splittable job can be interrupted (preempted) any number of times, and resume in the same mode after the interruption. A nonsplittable job is the job that must be performed continuously without interruption.

(2) At least one mode of operation is available for each job. Each mode reflects a different combination of resources and has a different job duration.

(3) A resource can be classified as one of three categories: renewable, nonrenewable and doubly-constrained.

(4) Resource consumption by each job and job duration corresponding to each mode are assumed to be discrete and deterministic.

(5) Activity-on-node diagram is used to represent the network, and looping of activities is not allowed.

(6) Job arrivals are known and take place at the end of discrete time periods.

(7) No scheduled or directed times are allowed on any intermediate job; i.e., the start and finish time of intermediate jobs are controlled only by factors such as precedence relationships and resource constraints.

(8) Notation based on working days rather than calendar days is used to represent time periods.

The notations and decision variables used in the mathematical model are listed in Table 3.1.

Constraints

1. Job Completion

For both splittable and nonsplittable jobs, each job can have only one completion period and use only one mode of operation, i.e, each job can be accomplished only once.

$$\sum_{m=1}^{M_j} \sum_{t=EF_{ij}}^{LF_{ij}} X_{ijmt} = 1$$

for $i = 1,\ldots,I$ and $j = 1,\ldots,N_i$

For splittable jobs,

(1) Each job can have only one start time and use only one mode of operation. Restarting a job after preemption is not considered as a start time.

TABLE 4.1 DEFINITION OF NOTATIONS AND DECISION VARIABLES

SYMBOL	DEFINITION
Notations	
i	project index $(i = 1,\ldots,I)$
I	number of projects
j	job index $(j = 1,\ldots,N_i)$
N_i	number of jobs in project i
L_i	set of jobs with no successors (or set of last jobs) for project i $(L_i \in L)$
L	set of jobs with no successors (or set of last jobs) for all projects
B	set of all pairs of immediate predecessor jobs
m	mode of operation index $(m = 1,\ldots,M_j)$
M_j	number of modes available for job j
k	renewable resource index $(k = 1,\ldots,K)$
K	number of different renewable resource types
p	nonrenewable resource index $(p = 1,\ldots,P)$
P	number of different nonrenewable resource types
t	time period $(t = 1,\ldots,\max G_i)$
G_i	absolute due date of project i (Project i must be completed in or before period G_i. If the absolute due date is not specified, G_i is the last period of the scheduling horizon.)
g_i	desired due date of project i (Project i is not late if it is completed in or before period g_i.)
l_i	penalty cost of project i per period of lateness (Assumed to be constant for all periods.)

TABLE 4.1 (continued)

SYMBOL	DEFINITION
c_i	overhead cost of project i per period
u_p	cost of nonrenewable resource type p per unit
Z_i	completion time of project i
Z	completion time of all projects
a_i	arrival period of project i (Arrivals occur at the end of periods.)
f_{ij}	number of periods preempted by job j, project i
h_{iab}	lead/lag time between jobs a and b, project i
d_{ijm}	number of periods required by mode m for job j, project i
ES_{ij}	earliest possible start time of job j, project i
EF_{ij}	earliest possible completion time of job j, project i
LS_{ij}	latest possible start time of job j, project i
LF_{ij}	latest possible completion time of job j, project i
r_{ijmk}	amount of type k renewable resource required by mode m for job j, project i
R_{kt}	amount of type k renewable resource available in period t
D_{kt}	desired level of renewable resource type k in period t (used for the resource leveling objective)
W_p	amount of nonrenewable resource type p available for all projects
W_{pt12}	amount of nonrenewable resource type p available for all projects between periods t1 and t2

TABLE 4.1 (continued)

SYMBOL	DEFINITION
w_{ijmp}	amount of nonrenewable resource type p required by mode m for job j, project i

Decision Variables

X_{ijmt}	= 1	if job j (splittable or non-splittable) of project i operating under mode m is completed at the end of period t.
	= 0	otherwise.
Y_{ijmt}	= 1	if splittable job j of project i operating under mode m is active during period t.
	= 0	otherwise.
S_{ijmt}	= 1	if splittable job j of project i operating under mode m is started at the end of period t (at the beginning of period t+1).
	= 0	otherwise.

NOTES: 1. For a doubly-constrained resource, $r_{ijmk} = w_{ijmp}/d_{ijm}$ where k and p are the same resource type.

2. To reduce the number of decision variables, the lower and upper bounds of each decision variable should be determined in advance. The bounds can be computed by using the time-based bounding technique as follows:

(a) Obtain the ES and LF for each job from the forward/backward computations by assuming that each job is performed under the shortest duration mode, and the latest finish time of each terminal activity is set to an absolute due date.

(b) The EF of job j mode m is given by the sum of ES obtained from Step (1) and the duration of job j mode m. The LS of job j mode m is computed from the difference between LF obtained from Step (1) and the duration of job j mode m.

$$\sum_{m=1}^{M_j} \sum_{t=ES_{ij}}^{LS_{ij}} S_{ijmt} = 1$$

for $i = 1,\ldots,I$ and $j = 1,\ldots,N_i$

(2) A splittable job cannot be completed unless the number of active periods ($\sum_q Y$) equals job duration ($\sum_t dX$, X equals one when the job is completed).

$$\sum_{t=EF_{ij}}^{LF_{ij}} d_{ijm} X_{ijmt} = \sum_{q=ES_{ij}+1}^{LF_{ij}} Y_{ijmq}$$

for $i = 1,\ldots,I$

$j \in \{\text{splittable jobs}\}$

$m = 1,\ldots,M_j$

2. Project Completion

A project is completed only when all jobs (N_i) in the project have been processed ($\sum_j\sum_m\sum_t X$).

$$\sum_{j=1}^{N_i} \sum_{m=1}^{M_j} \sum_{t=EF_{ij}}^{LF_{ij}} X_{ijmt} = N_i$$

for $i = 1,\ldots,I$

3. Precedence Relationships

For splittable jobs, a predecessor job, a, must be completed ($\sum_m\sum_t tX$) before a successor job, b, can start ($\sum_m\sum_t tS$) for all pairs of predecessor-successor jobs, B.

$$-\sum_{m=1}^{M_a} \sum_{t=EF_{ia}}^{LF_{ia}} tX_{iamt} + \sum_{m=1}^{M_b} \sum_{t=ES_{ib}}^{LS_{ib}} tS_{ibmt} \geq 0$$

for all pairs $(a,b) \in B$

$i = 1,...,I$

For nonsplittable jobs, a predecessor job, a, must be completed ($\sum_m \sum_t tX$) before a successor job, b, can start [$\sum_m \sum_t (t-d)X$] for all pairs of predecessor-successor jobs, B.

$$-\sum_{m=1}^{M_a} \sum_{t=EF_{ia}}^{LF_{ia}} tX_{iamt} + \sum_{m=1}^{M_b} \sum_{t=EF_{ib}}^{LF_{ib}} (t-d_{ibm}) X_{ibmt} \geq 0$$

for all pairs $(a,b) \in B$

$i = 1,...,I$

4. <u>Resource Constraints</u>

For renewable resources at each time period t, the total resource consumption at time period t by all nonsplittable jobs ($\sum_i \sum_j \sum_m \sum_q rX$) and by all splittable jobs ($\sum_i \sum_{j'} \sum_m rY$) must be less than or equal to the available amount of resource k at time period t (R_{kt}).

$$\sum_{i=1}^{I} \sum_{\forall j} \sum_{m=1}^{M_j} \sum_{q=t}^{t+d_{ijm}-1} r_{ijmk} X_{ijmq} + \sum_{i=1}^{I} \sum_{\forall j'} \sum_{m=1}^{M_{j'}} r_{ij'mk} Y_{ij'mt} \leq R_{kt}$$

for $j \in$ {nonsplittable jobs}

$j' \in$ {splittable jobs}

$k = 1,...,K$

$t = 1,...,\max G_i$

For nonrenewable resources, total resource consumption of all jobs ($\sum_i \sum_j \sum_m \sum_t wX$) may not exceed the available amount for renewable resource type p (W_p); d_p^- represents the under-utilization (slack variable) of resource p.

$$\sum_{i=1}^{I} \sum_{j=1}^{N_i} \sum_{m=1}^{M_j} \sum_{t=EF_{ij}}^{LF_{ij}} w_{ijmp} X_{ijmt} + d_p^- = W_p$$

for $p = 1,\ldots,P$

If it is desired to control the availability of nonrenewable resources within a given interval $[W_p(t1,t2)]$, the above constraint becomes:

$$\sum_{i=1}^{I} \sum_{j=1}^{N_i} \sum_{m=1}^{M_j} \sum_{q=t_1}^{t_2} w_{ijmp} X_{ijmq} + d_p^- = W_p(t_1,t_2)$$

for $p = 1,\ldots,P$

$(t1,t2) \in (1,\max G_i)$

5. <u>Specific Constraints for Splittable Jobs</u>

5.1) A splittable job must use the same mode of operation throughout the process. If a job starts by using mode m′ ($\sum_t S=1$), it must continue to use the same mode. If a job completes by using mode m ($\sum_m \sum_t X=1$), it cannot start by using mode m′ where m′ is different from m.

$$\sum_{t=ES_{ij}}^{LS_{ij}} S_{ijm't} + \sum_{\forall m} \sum_{t=EF_{ij}}^{LF_{ij}} X_{ijmt} = 1$$

for $i = 1,\ldots,I$

$j \in \{\text{splittable jobs}\}$

$m' = 1,\ldots,M_j$

$m \in \{M_j - m'\}$

5.2) The summation of start time and job duration $[\sum_m \sum_t (t+d)S]$ must be greater than or equal to the completion time ($\sum_m \sum_t tX$). This allows a splittable job to be preempted for f_{ij} periods.

$$\sum_{m=1}^{M_j} \sum_{t=ES_{ij}}^{LS_{ij}} (t+d_{ijm}) S_{ijmt} + f_{ij} = \sum_{m=1}^{M_j} \sum_{t=EF_{ij}}^{LF_{ij}} tX_{ijmt}$$

for $i = 1,\ldots,I$

$j \in$ {splittable jobs}

5.3) For control over active periods, an active period (the period that a splittable job is being processed) must lie between the start time and the completion time of the job.

(a) The active period [$(t-1)Y$] must be greater than or equal to the start time of the job ($\sum_q qS$).

$$(t-1)Y_{ijmt} \geq \sum_{q=ES_{ij}}^{LS_{ij}} qS_{ijmq}$$

for $i = 1,\ldots,I$

$j \in$ {splittable jobs}

$m = 1,\ldots,M_j$

$t = (ES_{ij}+1),\ldots,(LS_{ij}+1)$

(b) The active period (tY) must be less than or equal to the completion time of the job ($\sum_q qX$).

$$tY_{ijmt} \leq \sum_{q=EF_{ij}}^{LF_{ij}} qX_{ijmq}$$

for $i = 1,\ldots,I$

$j \in$ {splittable jobs}

$$m = 1, \ldots, M_j$$

$$t = EF_{ij}, \ldots, LF_{ij}$$

To assure that the first period a job is active is the same as the start time of the job, and the last period that the job is active is the same as the completion time of the job; the number of splitting period (f_{ij}) must be minimized.

6. Other Types of Dependency Relationship

In addition to the finish-to-start relationship without a lead/lag time, other types of dependency relationship with a lead/lag time can be formulated as defined below. Constraints (6.1) - (6.5) assume that job a precedes job b.

6.1) Finish-to-Start Relationship: the start time of job b ($\sum_m \sum_t tS$) must be greater than the finish time of job a ($\sum_m \sum_t tX$); h_{iab} represents the time difference between the start of b and ending of a.

$$\sum_{m=1}^{M_a} \sum_{t=EF_{ia}}^{LF_{ia}} tX_{iamt} + h_{iab} = \sum_{m=1}^{M_b} \sum_{t=ES_{ib}}^{LS_{ib}} tS_{ibmt}$$

6.2) Start-to-Start Relationship: the start time of job b ($\sum_m \sum_t tS_b$) must be greater than the start time of job a ($\sum_m \sum_t tS_a$); h_{iab} represents the time difference between the start of b and start of a.

$$\sum_{m=1}^{M_a} \sum_{t=ES_{ia}}^{LS_{ia}} tS_{iamt} + h_{iab} = \sum_{m=1}^{M_b} \sum_{t=ES_{ib}}^{LS_{ib}} tS_{ibmt}$$

6.3) Finish-to-Finish Relationship: the finish time of job b ($\sum_m\sum_t tX_b$) must be greater than the finish time of job a ($\sum_m\sum_t tX_a$); h_{iab} represents the time difference between the ending of b and ending of a.

$$\sum_{m=1}^{M_a} \sum_{t=EF_{ia}}^{LF_{ia}} tX_{iamt} + h_{iab} = \sum_{m=1}^{M_b} \sum_{t=EF_{ib}}^{LF_{ib}} tX_{ibmt}$$

6.4) Start-to-Finish Relationship: the finish time of job b ($\sum_m\sum_t tX$) must be greater than the start time of job a ($\sum_m\sum_t tS$); h_{iab} represents the time difference between the ending of b and start of a.

$$\sum_{m=1}^{M_a} \sum_{t=ES_{ia}}^{LS_{ia}} tS_{iamt} + h_{iab} = \sum_{m=1}^{M_b} \sum_{t=EF_{ib}}^{LF_{ib}} tX_{ibmt}$$

6.5) Composite Start-to-Start and Finish-to-Finish Relationship: constraints (6.2) and (6.3) are applied to the job simultaneously.

$$\sum_{m=1}^{M_a} \sum_{t=ES_{ia}}^{LS_{ia}} tS_{iamt} + h_{iab} = \sum_{m=1}^{M_b} \sum_{t=ES_{ib}}^{LS_{ib}} tS_{ibmt} \text{ , and}$$

$$\sum_{m=1}^{M_a} \sum_{t=EF_{ia}}^{LF_{ia}} tX_{iamt} + h_{iab} = \sum_{m=1}^{M_b} \sum_{t=EF_{ib}}^{LF_{ib}} tX_{ibmt}$$

Since the decision variable S_{ijmt} is not applicable to nonsplittable jobs, the term $\sum_{m=1}^{M_j} \sum_{t=ES_{ij}}^{LS_{ij}} tS_{ijmt}$ is replaced by $\sum_{m=1}^{M_j} \sum_{t=EF_{ij}}^{LF_{ij}} (t-d_{ijm}) X_{ijmt}$.

7. <u>Concurrency and Nonconcurrency of Completion Time</u>

7.1) Jobs a and b must be completed simultaneously.

This is true only if the completion time of job a ($\sum_m\sum_t tX_a$) equals the completion time of job b ($\sum_m\sum_t tX_b$).

$$\sum_{m=1}^{M_a} \sum_{t=EF_{ia}}^{LF_{ia}} tX_{iamt} = \sum_{m=1}^{M_b} \sum_{t=EF_{ib}}^{LF_{ib}} tX_{ibmt}$$

7.2) Nonconcurrency of jobs a and b. This is true only if the completion time of job a ($\sum_m\sum_t tX_a$) does not equal the completion time of job b ($\sum_m\sum_t tX_b$).

$$\sum_{m=1}^{M_a} \sum_{t=EF_{ia}}^{LF_{ia}} tX_{iamt} \neq \sum_{m=1}^{M_b} \sum_{t=EF_{ib}}^{LF_{ib}} tX_{ibmt}$$

8. Constraints for Time-Related and Cost-Related Objectives

8.1) Minimizing the total project throughput time objective. The completion time of project i (Z_i) must be greater than or equal to the completion time of all jobs in the set of last jobs for project i ($\sum_m\sum_t tX_j$, $\forall j \in L_i$).

$$Z_i \geq \sum_{m=1}^{M_j} \sum_{t=EF_{ij}}^{LF_{ij}} tX_{ijmt}$$

for all $j \in L_i$

$i = 1,\ldots,I$

8.2) Minimizing makespan objective. The completion time of all projects (Z) must be greater than or equal to the completion time of all jobs in the set of last jobs for all projects ($\sum_m\sum_t tX_j$, $\forall j \in L$).

$$Z \geq \sum_{m=1}^{M_j} \sum_{t=EF_{ij}}^{LF_{ij}} tX_{ijmt}$$

for all $j \in L$

9. Constraints for Cost-Related Objective and Time-Cost Tradeoffs Problem

The project must be completed (Z_i) by the due date (G_i).

$$Z_i \leq G_i$$

for $i = 1,\ldots,I$

10. Constraints for Resource Leveling Objective

10.1) The objective expresses the minimization of the absolute deviation between resource consumption and some desired level. The total renewable resource consumption at time period t by all nonsplittable jobs ($\sum_i\sum_j\sum_m\sum_q rX$) and by all splittable jobs ($\sum_i\sum_{j'}\sum_m rY$) must equal the desired level, D_{kt}; v_{kt}^- and v_{kt}^+ represent the under- and over-utilization of resource k at time period t from the desired level.

$$\sum_{i=1}^{I} \sum_{\forall j} \sum_{m=1}^{M_j} \sum_{q=t}^{t+d_{ijm}-1} r_{ijmk}X_{ijmq} + \sum_{i=1}^{I} \sum_{\forall j'} \sum_{m=1}^{M_{j'}} r_{ij'mk}Y_{ij'mt} - D_{kt} + v_{kt}^- - v_{kt}^+ = 0$$

where $v_{kt}^-, v_{kt}^+ \geq 0$

$j \in$ {nonsplittable jobs}

$j' \in$ {splittable jobs}

$k = 1,\ldots,K$

$t = 1,\ldots,\max G_i$

10.2) The objective expresses the minimization of the absolute deviation between resource consumption of consecutive time periods. The total renewable resource consumption at time period (t+1) by all nonsplittable jobs ($\sum_i \sum_j \sum_m \sum_{q=t+1} rX_q$) and by all splittable jobs ($\sum_i \sum_{j'} \sum_m rY_{t+1}$) is desired to be equal to the resource level at the previous time period, t. The deviation variables u_{kt}^- and u_{kt}^+ represent negative and positive deviations of resource levels from one period to the next.

$$\sum_{i=1}^{I} \sum_{\forall j} \sum_{m=1}^{M_j} \sum_{q=t+1}^{t+d_{ijm}} r_{ijmk} X_{ijmq} + \sum_{i=1}^{I} \sum_{\forall j'} \sum_{m=1}^{M_{j'}} r_{ij'mk} Y_{ij'm(t+1)}$$

$$- \sum_{i=1}^{I} \sum_{\forall j} \sum_{m=1}^{M_j} \sum_{q=t}^{t+d_{ijm}-1} r_{ijmk} X_{ijmq} - \sum_{i=1}^{I} \sum_{\forall j'} \sum_{m=1}^{M_{j'}} r_{ij'mk} Y_{ij'mt} + u_{kt}^- - u_{kt}^+ = 0$$

where u_{kt}^-, $u_{kt}^+ \geq 0$

$j \in$ {nonsplittable jobs}

$j' \in$ {splittable jobs}

$k = 1, \ldots, K$

$t = 1, \ldots, (\max G_i) - 1$

Objective Functions

1. Minimizing Total Project Throughput Time

Since project throughput time is given by the difference between the project completion period (Z_i) and the arrival period (a_i), the objective function can be written as:

$$\text{Minimize} \sum_{i=1}^{I} Z_i$$

If it is desired to start all jobs as soon as possible without increasing throughput time, all nonsplittable jobs, j, must be completed as soon as possible (Min $\sum_i\sum_j\sum_m\sum_t tX$) and all splittable jobs, j', must start as soon as possible (Min $\sum_i\sum_{j'}\sum_m\sum_t tS$).

$$\text{Minimize} \quad \sum_{i=1}^{I} \sum_{\forall j} \sum_{m=1}^{M_j} \sum_{t=EF_{ij}}^{LF_{ij}} tX_{ijmt} + \sum_{i=1}^{I} \sum_{\forall j'} \sum_{m=1}^{M_{j'}} \sum_{t=ES_{ij'}}^{LS_{ij'}} tS_{ij'mt}$$

for $j \in$ {nonsplittable jobs}

$j' \in$ {splittable jobs}

If it is desired to start all jobs as late as possible without increasing throughput time, the difference between the project completion period (Z_i) and the completion period of each job (tX) must be minimized.

$$\text{Minimize} \quad \sum_{i=1}^{I} \sum_{j=1}^{N_i} \sum_{m=1}^{M_j} \sum_{t=EF_{ij}}^{LF_{ij}} (Z_i - tX_{ijmt})$$

2. <u>Minimizing Makespan</u>

The makespan can be minimized when the completion time of all projects (Z) is minimized.

$$\text{Minimize} \quad Z$$

If it is desired to start all jobs as soon as possible without increasing makespan, all nonsplittable jobs (j) must be completed as soon as possible (Min $\sum_i\sum_j\sum_m\sum_t tX$) and all splittable jobs (j') must start as soon as possible (Min $\sum_i\sum_{j'}\sum_m\sum_t tS$).

$$\text{Minimize} \quad \sum_{i=1}^{I} \sum_{\forall j} \sum_{m=1}^{M_j} \sum_{t=EF_{ij}}^{LF_{ij}} tX_{ijmt} + \sum_{i=1}^{I} \sum_{\forall j'} \sum_{m=1}^{M_{j'}} \sum_{t=ES_{ij'}}^{LS_{ij'}} tS_{ij'mt}$$

for $j \in$ {nonsplittable jobs}

$j' \in$ {splittable jobs}

If it is desired to start all jobs as late as possible without increasing makespan, the difference between the completion period of all projects (Z) and the completion period of each job (tX) must be minimized.

$$\text{Minimize} \quad \sum_{i=1}^{I} \sum_{j=1}^{N_i} \sum_{m=1}^{M_j} \sum_{t=EF_{ij}}^{LF_{ij}} (Z - tX_{ijmt})$$

3. <u>Minimizing Total Lateness or Lateness Penalty</u>

A project is late if $Z_i > g_i$. The total project lateness can be defined as $\sum_{i=1}^{I} (Z_i - g_i)$ and the total lateness penalty can be defined as $\sum_{i=1}^{I} l_i (Z_i - g_i)$. The objective function of lateness penalty then becomes:

$$\text{Minimize} \quad \sum_{i=1}^{I} l_i (Z_i - g_i)$$

The objective function of lateness penalty can be reduced to the objective function of total project lateness when all l_i are equal to one.

4. <u>Minimizing Project Overhead Cost</u> (Indirect Cost)

The project overhead cost can be defined as the product of overhead cost per period (c_i) and project completion time (Z_i).

$$\text{Minimize} \quad \sum_{i=1}^{I} c_i Z_i$$

5. <u>Minimizing Project Direct Cost</u>

5.1) Direct cost due to renewable resources

Let w_{ijmc} be the total cost (nonrenewable resource type c) associated with all renewable resources used in job j of project i operating in mode m. The objective of minimizing project direct cost of all jobs ($\sum_i\sum_j\sum_m\sum_t wX$) can be given by:

$$\text{Minimize} \quad \sum_{i=1}^{I} \sum_{j=1}^{N_i} \sum_{m=1}^{M_j} \sum_{t=EF_{ij}}^{LF_{ij}} w_{ijmc} X_{ijmt}$$

5.2) Direct cost due to nonrenewable resources

To calculate direct cost associated with nonrenewable resources, the slack variable d_p^- should be introduced in the nonrenewable resource constraint (Constraint 4). Since the total resource consumption equals (W_p-d_p^-), the total cost of nonrenewable resources is equal to $u_p(W_p-d_p^-)$ and the objective function becomes:

$$\text{Minimize} \quad \sum_{p=1}^{P} u_p (W_p - d_p^-)$$

The indirect cost objective (project overhead) in (4) and the direct cost objectives in (5) can be combined to obtained the total cost of project,

$$\text{Minimize} \quad \sum_{i=1}^{I} c_i Z_i + \sum_{i=1}^{I} \sum_{j=1}^{N_i} \sum_{m=1}^{M_j} \sum_{t=EF_{ij}}^{LF_{ij}} w_{ijmc} X_{ijmt} + \sum_{p=1}^{P} u_p (W_p - d_p^-)$$

6. <u>Resource Leveling Objectives</u>

6.1) Minimizing the absolute deviation between resource consumption and a desired rate. The total amount of resource underutilized (v_{kt}^-) and overutilized (v_{kt}^+) as given in constraint (10.1) must be minimized,

$$\text{Minimize} \quad \sum_{k=1}^{K} \sum_{t=1}^{\max G_i} (v_{kt}^- + v_{kt}^+)$$

6.2) Minimizing the absolute deviation between resource consumption of consecutive time periods. The total amount of resource underutilized (u_{kt}^-) and overutilized (u_{kt}^+) as given in constraint (10.2) must be minimized,

$$\text{Minimize} \quad \sum_{k=1}^{K} \sum_{t=1}^{\max G_i - 1} (u_{kt}^- + u_{kt}^+)$$

7. <u>Multiple Objective Function</u>

A preemptive goal programming model can be formulated by adding the goal constraint associated with each goal in addition to the real constraints, and using the deviation variables in the objective functions. Assume that a priority level, P_i (i = 1,...,k), is assigned to goal i, and $P_i >>> P_{i+1}$. T_i is the target level of goal i, and d is the deviation variable associated with the goal constraint, $G_i()$. The model can be written as:

$$\mathit{Minimize} \quad \sum_{i=1}^{k} P_i (d_i^+ + d_i^-)$$

subject to:

$$G_i() - d_i^+ + d_i^- = T_i$$

$$d_i^+, d_i^- \geq 0, \quad i = 1,...,k$$

4.2 Experience with the Mathematical Models

Mathematicians [Karp (1975); Lenstra and Rinnooy (1984)] refer to the resource-constrained project scheduling problem as a NP-complete combinatorial problem which is computational impractical to solve for most real-life problems. Many integer programming formulations had been developed in the past, but none of them seem to show a successful result in term of computation time. These include the time minimization model by Wiest (1963), the cost minimization model by Hadley (1964), the multi-resource time minimization by Brand et al. (1964), and the resource leveling problem by Moodie and Mandeville (1965). Balinski (1965) reported that the integer programming approach was successful only to covering type problems, but not combinatorial problems such as scheduling problems. Moodie and Mandeville (1965) commented that even though there exists an efficient approach for solving the integer programming, it still requires a tremendous amount of effort to write all objectives functions and constraints of the problem. Even though the efficient zero-one programming formulation such as the Pritsker, Watters and Wolfe (1969) model was developed to reduce the size of the problem, it is still unsuccessful in term of applications as reported by Hall (1980).

Solving mathematical models through integer programming approach seem to be inefficient. Approaches such as implicit enumeration and branch-and-bound have received more attention during the past years. By utilizing the special structure in a problem of interest, the time required to obtain optimal solutions can be greatly reduced. The three prominent approaches include the techniques developed by Davis and Heidorn (1971), Stinson et al. (1978), and Talbot and Patterson (1978). Patterson (1984) made a comparison among these techniques, and gave some suggestions about the extension of these techniques to include job splitting, variation in resource usage, and variation in resource availabilities. Talbot's algorithm seems to benefit from less computer memory requirement and reasonable computation time, and it is also easier to include the extensions stated earlier. Thus, the technique developed by Talbot (1978, 1982) will be used as a basis for developing the proposed single objective algorithms. The concept of implicit enumeration for zero-one goal programming introduced by Lee (1979) will then be used in the multiple objective algorithm. These concepts are explained in the next chapter.

CHAPTER 5

SOLUTION ALGORITHMS

5.1 Talbot's Algorithm

Talbot's time minimization algorithm for resource-constrained project scheduling is a two-stage approach which includes the labeling process and the enumeration process. The labeling process is performed at the first stage to specify the order of jobs, modes, and resources to be considered in the next stage. The enumeration process attempts to systematically evaluate all possible scheduled times for each job in the project. During the process, the augmentation process attempts to assign a new feasible job to a partial schedule, and the backtracking process attempts to reschedule a job previously assigned to the partial schedule. Optimality is reached when all jobs are systematically evaluated. Details of the algorithm are given below.

Stage 1 Labeling Process

1. Job Labeling Process: The job labeling process is primarily used to determine the order in which jobs will be assigned to the schedule in the following stage. In addition, the precedence relationship can be maintained by using the rearranging process. First, jobs are labeled according to the job labeling rule selected from the list given in Table 5.1. Instead of checking for the feasibility of precedence relationship in the next stage, the rearranging process is

TABLE 5.1 JOB LABELING RULES

RULE	DEFINITION
1. MAAD	maximum average job duration $$\sum_{m=1}^{M_j} d_{jm}/M_j$$
2. MAD	maximum job duration $$\text{Max } \{d_{jm} \mid m = 1, \ldots, M_j\}$$
3. MAAR	maximum average resource demand $$[\sum_{m=1}^{M_j} d_{jm}/M_j] * [\sum_{m=1}^{M_j} \sum_{k=1}^{K} r_{jmk}/M_j K]$$
4. MIE	minimum earliest finish time (EF_j)
5. MIL	minimum latest finish time (LF_j)
6. MILSD	minimum of the difference between the latest finish time and the smallest duration $$[LF_j - \min \{d_{jm} \mid m = 1, \ldots, M_j\}]$$
7. MILAD	minimum of the difference between the latest finish time and the average duration $$[LF_j - \sum_{m=1}^{M_j} d_{jm}/M_j]$$
8. RAN	jobs are numbered randomly

<u>NOTES</u> (1) The above job labeling rules are the same rules used by Talbot (1982). For details, see Talbot (1982, Table 1 page 1208).

(2) The EF and LF are based on the shortest duration mode of each job.

used in this stage to reduce the computation time and memory requirements. Jobs are rearranged such that the predecessor jobs must have lower labeling numbers than the successor jobs while maintaining the order obtained from the selected labeling rule as much as possible.

2. Mode Labeling Process: The process will order modes of each job according to the selected mode labeling rule which itself depends on the objective function. If the objective is time minimization, modes are labeled in increasing order of duration. If the objective is cost minimization, modes are labeled in increasing order of total cost. If the objective is resource leveling, modes are labeled in increasing order of total usage of renewable resources.

3. Resource Labeling Process: This process can reduce the computation time required for identifying infeasible resources in the next stage. Renewable resources are labeled in decreasing order of the index representing the frequency of highest per-period resource requirement relative to average resource availability. The index is defined as follows:

$$Index_k = \sum_{\forall j} \sum_{\forall m} I_{jkm}$$

where I_{jkm} = d_{jm} for the k which maximizes the ratio of r_{jkm} to $\overline{R}_k$

= 0, otherwise

$$\overline{R}_k = \frac{\sum_{i=1}^{p} \Delta t_i R_{kt_i}}{\sum_{i=1}^{p} \Delta t_i}$$

p = number of intervals that resource type k varies

Δt_i = time within interval i

R_{kt_i} = number of units available for resource type k during interval i.

Stage 2 Enumeration Process

The objective of Stage 2 is to find the complete schedule which optimizes the desired objective function. A complete schedule is built up from partial schedules in the enumeration process. A partial schedule is a feasible schedule, a schedule in which jobs satisfy all the constraints, but contains less than the total number of jobs. The enumeration process consists of two major steps, the augmentation process and the backtracking process.

The augmentation process is the process of adding a new feasible job to the partial schedule currently being considered. A feasible job is one that satisfies all the constraints within its range (earliest completion time to latest completion time). A complete schedule results if all jobs satisfy all constraints.

The backtracking process is the process of rescheduling a job previously assigned in the partial schedule. It tries to find an improved solution if one exists; otherwise, it is used to prove that the current solution is optimal.

First, the jobs are retrieved from Stage 1, and are assigned to a partial schedule in increasing order based on the selected job labeling rule. Jobs are assigned to the partial schedule at their earliest feasible completion times, the earliest time that a job can be completed without violating any constraint. If a job cannot be assigned within its range, an attempt is made to reassign the job by using a new mode. If the job cannot be assigned within its range by using any mode, the job is said to be unassigned and the backtracking process begins. The job immediately assigned before the unassigned job is selected for rescheduling at the time after its currently scheduled time. The process continues systematically until a complete schedule with an improved solution is found, or until optimality is reached. After an improved solution is found, the value of the objective function is tightened and the augmentation process resumes at job 1. Optimality is reached when either the backtracking process is made past job one, or the improved solution is found to be equal to a proven theoretical bound such as the critical path early completion time.

5.2 Improvements and Extensions of Talbot's Algorithm

Even though Talbot's algorithm represents significant improvements in terms of computational efficiency over mathematical optimization approaches, the algorithm is still time intensive and limited to problems with certain features. The following improvements and extensions were developed for the Talbot's algorithm.

1. Skip Mode Process

In the enumeration process, certain modes for a job can be eliminated from consideration if they cannot yield a better result than the currently scheduled mode. The skip mode process is applicable to problems for which the values of the objective function are computed from the sequential sum of the corresponding values for each job, e.g., total project cost is the sum of total cost of all jobs in the project. Thus, the skip mode process can be applied to cost minimization problems. In addition, it can also be used in time minimization and resource leveling problems if jobs are serially scheduled to the network, i.e., no more than one job is allowed to be scheduled at the same time. With the use of an appropriate mode labeling rule, the skip mode process becomes quite effective.

In cost minimization problems, modes that require higher total cost than the currently scheduled mode of the same job can be eliminated if the current mode has a cost bound beyond the desired project total cost. The cost bound of job j mode m is defined as the sum of the cumulative cost of all assigned jobs in the partial solution (jobs 1 to j-1), the cost of job j mode m, and the cumulative minimum cost of all unassigned jobs (jobs j+1 to the last job). Since modes are ordered in increasing total cost in the cost minimization objective, modes that have higher mode numbers than the current mode number can be eliminated from consideration.

2. <u>Bound Checking Process</u>

The bound checking process is designed to handle nonrenewable resource conflicts more effectively by performing a feasibility test of the nonrenewable resource bound for the intended mode of each job. The process attempts to find the earliest scheduled time of each job within its range that satisfies all the constraints, and has the nonrenewable resource bound less than or equal to the nonrenewable resource limit. If the currently scheduled mode of the job does not satisfy the above constraints, the next mode is considered for assignment. Every mode of a job must be considered because modes are not labeled in any order of nonrenewable resources; or if they are, they can only be ordered according to one type of nonrenewable resources.

In the bound checking process, the nonrenewable resource bound is used to check the feasibility of nonrenewable resources for the intended mode of a job instead of comparing the nonrenewable resource consumption of the intended mode to the amount of nonrenewable resource available as used in Talbot's algorithm. The nonrenewable resource bound of job j mode m is defined as the sum of the cumulative nonrenewable resource usage of all assigned jobs in the partial schedule (jobs 1 to j-1), the nonrenewable resource usage of job j mode m, and the cumulative minimum nonrenewable resource usage of all unassigned jobs (jobs j+1 to the last job). Even though the idea of the bound checking process is similar to the cost bound described in the previous section, the concept is broader and applicable to any type of nonrenewable resource, and also to any type of objective function.

3. Flexible Backtracking Process

To increase the flexibility and applicability of the algorithm, different backtracking rules are investigated. In Talbot's algorithm, the LIFO (last-in-first-out) concept is used to select a job for rescheduling in the backtracking process.

> " The depth-first search strategy embodied in the LIFO backtracking rule has the obvious advantage of simplicity, low storage requirement, and ease of bookkeeping.... On the other hand, the rigidity of the LIFO rule often becomes cumbersome, since it implies that a bad decision made early in the search cannot be corrected until late in the search. A strategy which preserves most of the advantages of the depth-first approach while getting rid of the rigidity of LIFO, is to augment the partial solution last generated whenever it cannot be fathomed; but when it can be fathomed, then to continue the search with the most promising unfathomed partial solution rather than with the immediate predecessor of the current one.... Finding a good criterion for selecting the node (partial solution) for a flexible backtrack is a matter of experimentation." (Balas, 1979 in Operations Research Support Methodology, pp. 476-478.)

The idea of flexible backtrack is implemented in the proposed algorithms. Before backtracking can begin, an unassigned job must first be determined. A job is said to be unassigned when one of the following events occurs:

1. During the augmentation process, an unassigned job is the job that cannot be assigned to a partial schedule within its range by using any mode. The unassigned job for each objective can be determined as follows:

(a) Time Minimization Problem: The unassigned job is the job that cannot be scheduled within its range by using any mode due to resource infeasibility.

(b) Cost Minimization Problem: The unassigned job is the job that cannot be scheduled within its range by using any mode due to resource infeasibility, or has its cost bound greater than the current project total cost.

(c) Resource Leveling Problem: The unassigned job is the job that cannot be scheduled within its range by using any mode due to resource infeasibility. Since the value of total resource deviation cannot be determined unless the schedule is completed, the value of total resource deviation cannot be compared to the tightened objective value until the last job is assigned to the schedule.

2. After a complete schedule with an improved solution is found, the objective value is tightened and an unassigned job is sought. The determination of an unassigned job from a complete schedule is as follows:

(a) Time Minimization Problem: In Talbot's algorithm, the project completion time and the latest finish time of each job are tightened after an improved solution is found, and the augmentation process starts at job 1, mode 1. The augmentation process continues until an unassigned job is determined. The computation time can be reduced if the unassigned job can be determined without restarting the augmentation process at job 1, mode 1. This can be accomplished by making a comparison between the tightened latest finish time and the completion time of all jobs in the incumbent complete schedule (the complete schedule that is most recently obtained). If a job's completion time is greater than the tightened latest finish time, then the job is a new unassigned job. If more than one job cannot be assigned, then the job with the lowest job number is selected. After an unassigned job is determined, an attempt is made to reschedule the unassigned job by using the next mode. If the unassigned job cannot be reassigned within its range by using any mode, then the backtracking process is initiated.

(b) Cost Minimization Problem: After an improved solution is found, the total project cost is tightened by a certain amount depending on a required accuracy (a deviation from optimal solution). Like the time minimization problem, Talbot's algorithm starts the augmentation process at job 1, mode 1, and continues until an unassigned job is determined. The following feasibility test is used to reduce the computation time to determine the unassigned job. Let j be the job that is currently tested; one of the following feasibility tests can then be used: (i) check that the cumulative cost from job 1 to job j is less than or equal to the tightened total project cost, or (ii) check that the cost bound of job j is less than or equal to the tightened total project cost. The use of cost bound approach is recommended because it gives a tighter bound which will result in less computation time. The following technique can then be used to determine an unassigned job without restarting the augmentation process at job 1, mode 1. An unassigned job is one whose cost bound is greater than the tightened total project cost. If more than one job satisfy this condition, the job with the lowest job number is selected. After an unassigned job is determined, the backtracking process begins. The other modes of the unassigned job can be eliminated from consideration because modes are ordered in increasing total cost.

(c) Resource Leveling Problem: Since the value of total resource deviation is obtained when a complete schedule is found, only the last job can be considered for rescheduling. After an improved solution is found and the objective value is tightened, the augmentation process begin at the currently scheduled mode of the last job in the incumbent solution. The process checks the feasibility of the last job at time after the currently scheduled finish time in the incumbent solution. If the last job cannot be scheduled within its range by using any mode, then the backtracking process is started.

If job j cannot be assigned to a schedule by using any mode, one of the jobs previously assigned in front of job j is considered for rescheduling according to one of the backtracking rules given in Table 5.2. If the selected job can be assigned at a later time within its range or by using another mode, then the augmentation process is started on the next job. If not, a new job is selected from the same set based on the rule previously used. The process continues systematically until an improved solution is found or optimality is reached. Optimality is reached when the backtracking process is made past job one. In flexible backtracking process, only the LIFO backtracking rule can guarantee optimality when the backtracking process is made past job one. The other backtracking rules may not yield

TABLE 5.2 BACKTRACKING RULES

RULE	DEFINITION[1]
1. LIFO	choose the most recently assigned job
2. MILF	choose the job with the smallest late finish time
3. MALF	choose the job with the largest late finish time
4. MITF	choose the job with the smallest total float
5. MATF	choose the job with the largest total float
6. MID	choose the job with the minimum duration
7. MAD	choose the job with the maximum duration
8. MIRD	choose the job with the smallest absolute resource deviation
9. MARD	choose the job with the largest absolute resource deviation
10. MITR	choose the job with the smallest total resource demand
11. MATR	choose the job with the largest total resource demand
12. RAN	choose the job randomly
13. MITC[2]	choose the job with the smallest total cost
14. MATC[2]	choose the job with the largest total cost

<u>NOTES</u> (1) If more than one job has the same value, choose the job with the highest job number.

(2) The MITC and MATC backtracking rules are applicable only to cost minimization problems.

optimal solutions because some job combinations may not be considered when a job is selected for rescheduling.

The algorithm is extended to search for optimal solutions or to guarantee optimality when other backtracking rules are used. This can be accomplished in two steps: (1) obtain a heuristic solution from these rules and use the solution as an initial upper bound for the objective function value, and (2) restart the enumeration process using the LIFO rule as a selected backtracking rule. The algorithm allows the user to specify if optimal solutions or heuristic solutions are desired when using a backtracking rule.

4. Job Splitting

It is assumed in this study that jobs can be split (preempted) any number of times. For the case where jobs can be split only at certain periods, see Patterson (1984) and Prisker et al. (1969). The algorithms handle splittable jobs differently from nonspilttable jobs in several ways, including:

(1) Resource availability is updated differently.

(2) Active time variables are used for splittable jobs instead of completion time variables.

(3) A splittable job is said to be completed when the number of active periods is equal to the job duration.

(4) To guarantee optimality, all possible combinations of active time variables of each splittable job

must be considered. Assume that a splittable job has a duration of 3 units. Then the active time variable for each time unit can be represented by Y1, Y2 and Y3, respectively. If the earliest start is at period 1 and the latest finish is at period 5, all possible combinations of active time variables are shown in Table 5.3.

TABLE 5.3

ACTIVE TIME COMBINATIONS FOR SPLITTABLE JOBS

TIME PERIOD				
1	2	3	4	5
Y1	Y2	Y3	-	-
Y1	Y2	-	Y3	-
Y1	Y2	-	-	Y3
Y1	-	Y2	Y3	-
Y1	-	Y2	-	Y3
Y1	-	-	Y2	Y3
-	Y1	Y2	Y3	-
-	Y1	Y2	-	Y3
-	Y1	-	Y2	Y3
-	-	Y1	Y2	Y3

A summary of the algorithm for nonsplittable jobs is given in Figure 5.1.

5. Resource Leveling Problem

In the resource leveling problem, the project completion time is fixed at a specified value, and jobs are scheduled to satisfy the resource leveling objective. First the earliest finish and the latest finish times of each job are calculated

FIGURE 5.1 FLOW CHART FOR A SPLITTABLE JOB

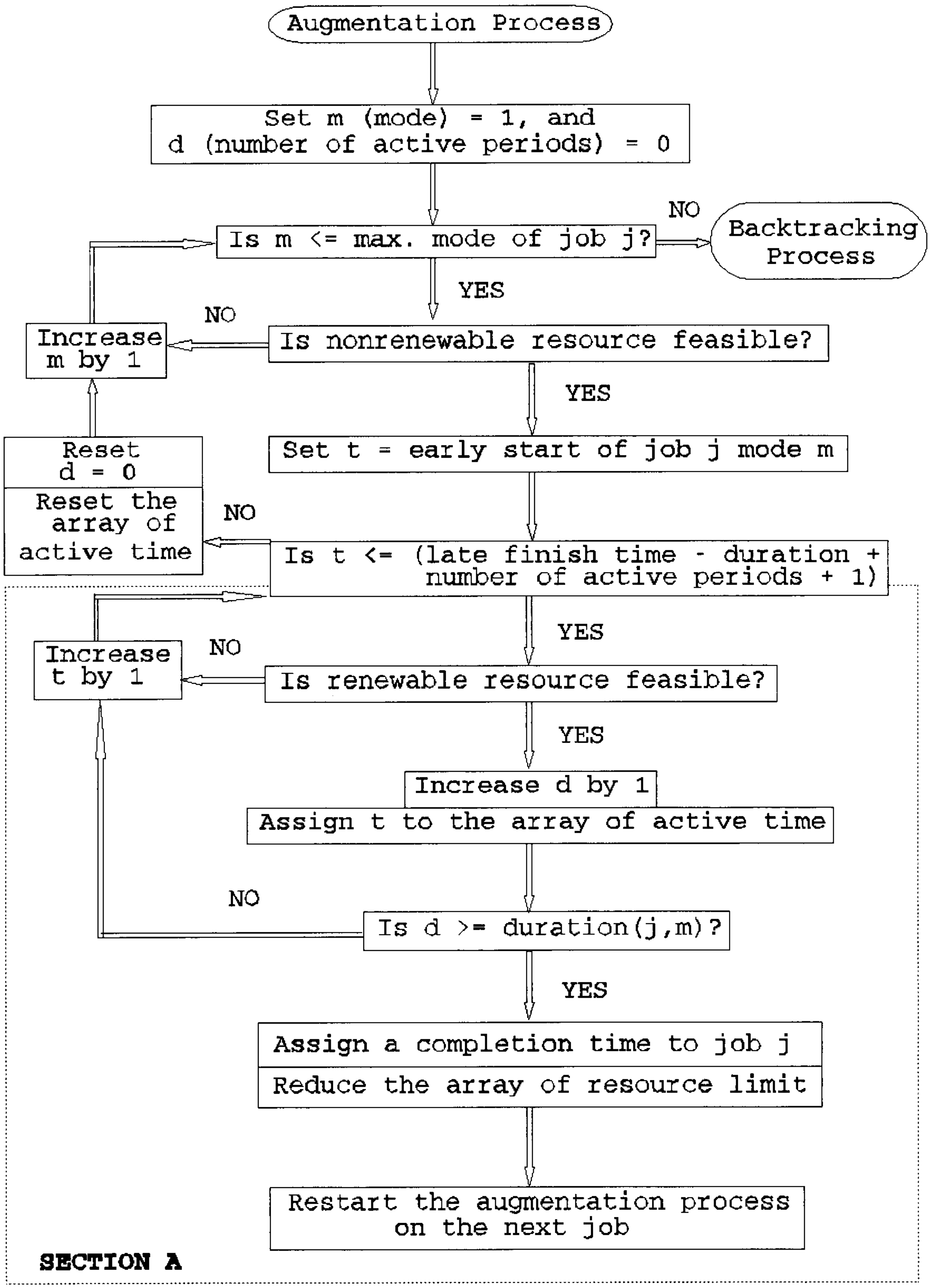
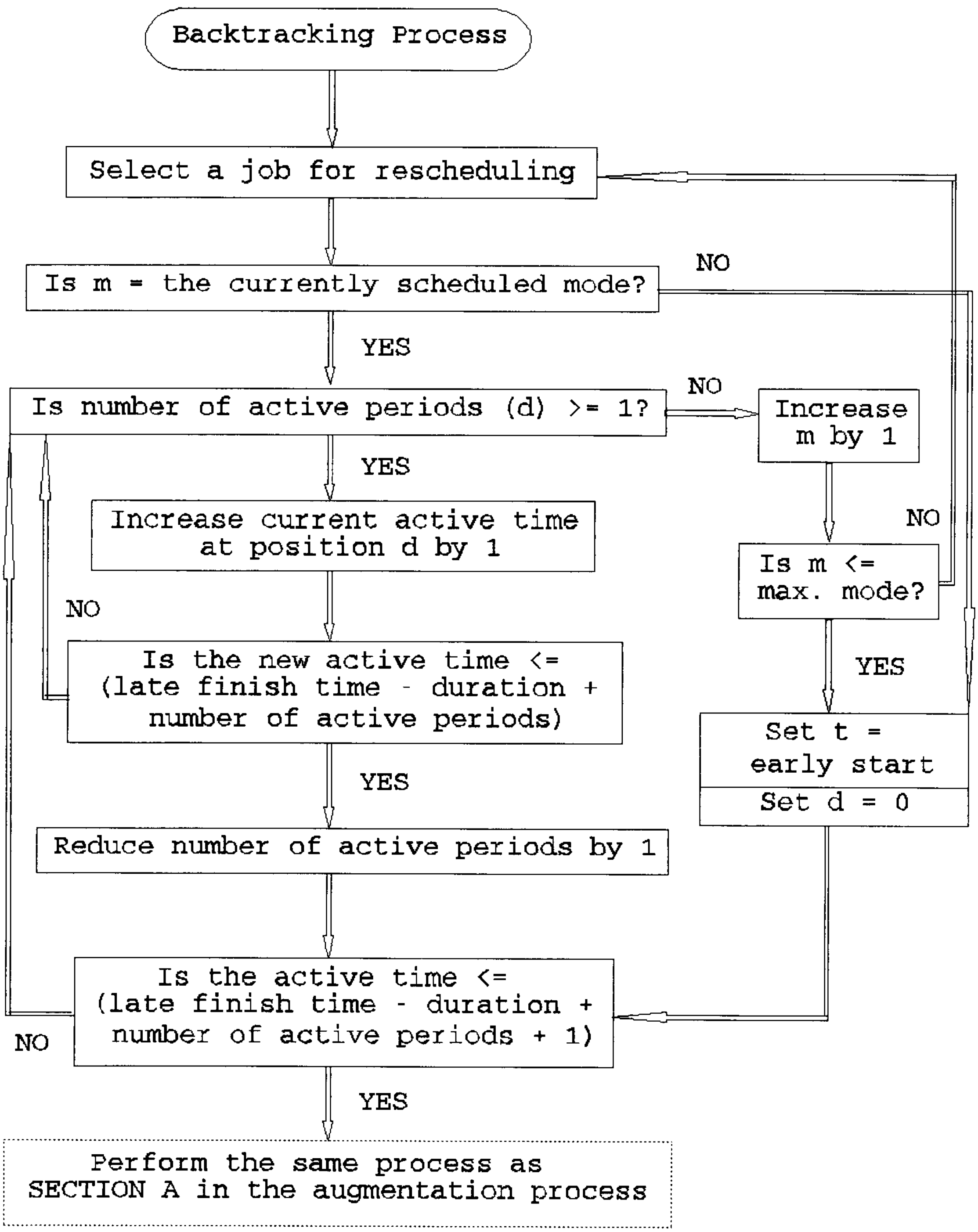


FIGURE 5.1 (continued)



based on the given completion times. The range remains unchanged since the project completion time is fixed. Two objectives frequently considered in resource leveling problems are: (1) to minimize the absolute deviation between resource consumption and a desired rate, and (2) to minimize the absolute deviation between the resource consumption of consecutive time periods.

The procedure for minimizing the absolute deviation from a desired rate is as follows:

(1) An incumbent objective value (D^*) is set to a known heuristic value. If a heuristic value is not known, D^* may be computed from

$$\sum_{\forall t} |\text{desired rate - resource used}|$$

when all jobs are assigned sequentially.

(2) The augmentation process begins by assigning jobs at their earliest feasible completion time and checking for resource feasibility. If all jobs are assigned to the schedule, the objective value (D) is computed. The objective value (D) is compared with the current incumbent objective value (D^*). If D is less than or equal to D^*, an improved solution is found (set the new incumbent objective value D^* to D). The value D^* is then reduced by 1 (called a tightened objective value), and the last job is scheduled by using the currently scheduled mode in the incumbent solution. If the job cannot be assigned a completion time within its given range, an attempt is made by using another mode. If the process is

still not successful, the flexible backtracking process begins. A job is selected by using one of the backtracking rules given in Table 5.2. If the selected job can be rescheduled within its range by using any mode, the augmentation process begins. If this is not possible, a new job is selected based on the same rule. The process continues systematically until an improved solution is found, or optimality is reached.

The same procedures can also be applied to the objective of minimizing the absolute deviation between consecutive time units. However, D and D^* now represent

$$\sum_{\forall t} |\text{resource usage at time } t - \text{resource usage at time } t\text{-}1|.$$

In this research, the desired rate is set equal to the renewable resource limit; thus, the problem becomes a resource-constrained version of the leveling problem. The advantages of this assumption are that the best level of resources can be determined at a planning stage, and resource utilization of the resource may be higher.

6. Multi-project Problem

The algorithms can be modified to handle a multi-project resource-constrained project scheduling problem by combining jobs from all projects into a single project and imposing additional due dates for some jobs or projects. The same process as the single project problem can be used to optimize the objective function value with some modification. For example, to minimize makespan, an attempt will be made to minimize the completion time of all terminal jobs from all projects.

7. <u>Multiple Objective Goal Programming Problem</u>

The multiple objective goal programming algorithm is developed for the time, cost, and resource leveling objectives. The idea is to find a solution that minimizes absolute deviations from a desired level for each objective without changing the objective values of higher priority goals. If the desired levels are unknown, a special algorithm is developed to sequentially optimize all objectives by finding the optimal values for all goals without changing the optimal values of higher priority goals.

In the proposed algorithm for the multiple objective goal programming problem, a goal constraint corresponding to the current goal is added, and real constraints corresponding to higher priority goals are imposed on the problem. The modified problem is then solved as a single objective problem. If the desired level of the current goal is met, the algorithm proceeds to the next lower priority goal. Otherwise, it attempts to find the solution that minimizes deviation from the desired level of the current goal. This process continues systematically until all goals have been considered and a solution is found. If the desired level is not specified, the desired level is set equal to an upper bound value or a known value obtained from the immediate higher priority goal. The desired level is tightened until the problem is no longer feasible. The solution before infeasibility is selected for the goal, and the algorithm proceeds to the next lower priority goal. In essence, two different approaches are applied to the multiple objective goal programming algorithm. The lower bound approach is used to handle the goal with a known desired level, and the upper bound approach is used to handle the goal with an unknown desired level.

5.3 Computerized Algorithms

Separate algorithms were developed for the single objective problems. A summary of the single objective algorithms is shown in Figure 5.2. A comparison among the algorithms for time minimization, cost minimization, and resource leveling is given in Table 5.4. In addition, the multiple objective algorithm was separately developed to handle the multiple objective resource-constrained project scheduling problem. The example shown in Figure 5.3 will be used to explain the computerized algorithms. The network consists of 4 jobs, and each job can be performed in two modes. There is one type of renewable resource with a limit of 7 units, and one type of nonrenewable resource representing total cost with a limit of $200. The resource deviation is the sum of the difference between the consumption level and the limit of renewable resource over the period of job duration in that mode.

FIGURE 5.2 SUMMARY OF THE SINGLE OBJECTIVE ALGORITHMS

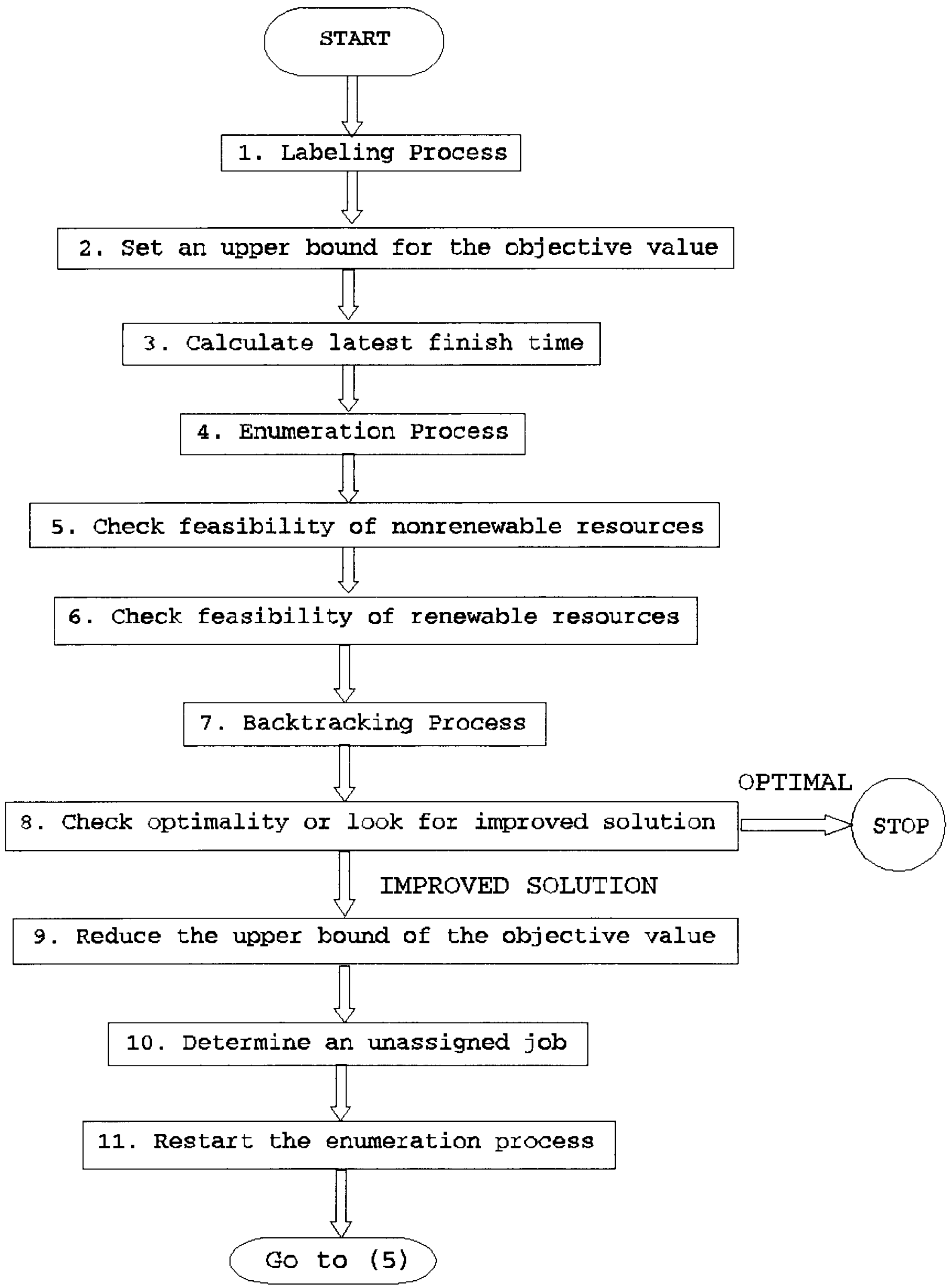
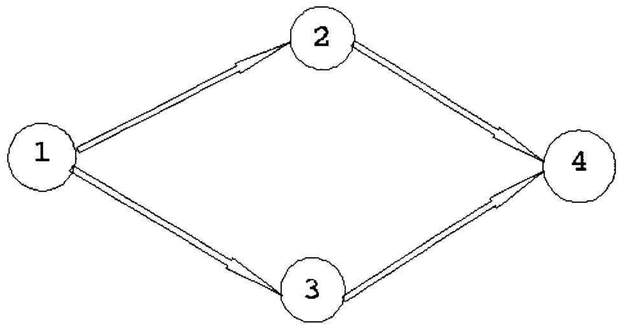


TABLE 5.4 COMPARISON OF SINGLE OBJECTIVE ALGORITHMS

STEP#	TIME	COST	RESOURCE LEVELING
1	Labeling Process	same as TIME	same as TIME
2	Set an upper bound for project completion time	Set an upper bound for project total cost	Set an upper bound for total resource deviation
3	Calculate latest finish time of each job based on the upper bound value	Calculate latest finish time of each job based on a desired completion time	same as COST
4	Augment one job at a time until the last job	Before augmenting any job, check feasibility of the total cost bound	same as TIME
5	Check feasibility of nonrenewable resources	same as TIME	same as TIME
6	Check feasibility of renewable resources	same as TIME	same as TIME
7	Backtracking Process	same as TIME	same as TIME
8	Check optimality of the current solution, or look for an improved solution	same as TIME	First, check feasibility of total resource deviation of the last job. Then same as TIME.
9	Reduce project completion time after improved solution is found	Reduce project total cost after improved solution is found	Reduce total resource deviation after improved solution is found
10	Reduce latest finish time of each job and compare to the incumbent values	Compare total cost bound of each job to tightened project total cost	Use the last job as an unassigned job
11	Begin enumeration process on the unassigned job	same as TIME	same as TIME

FIGURE 5.3 EXAMPLE PROBLEM



Job	Mode	Duration	Renewable Resource	Nonrenewable Resource	Resource Deviation
1	1	3	5	30	6
1	2	4	3	40	16
2	1	3	5	20	6
2	2	4	2	40	20
3	1	2	7	10	0
3	2	3	4	30	9
4	1	4	4	40	12
4	2	5	3	50	20

Renewable resource limit = 7

Nonrenewable resource (total cost) limit = 200

5.3.1 Time Minimization Algorithm

The solution procedure for time minimization problems is described in the following steps:

(1) Perform the labeling process.

(2) Set an incumbent project completion time (T^*) equal to a known heuristic value. If a heuristic value is not known, set T^* equal to the sum of all maximum job durations.

(3) Calculate the latest finish time of each job by using the shortest duration mode and the project completion time of T^*.

(4) Let X_j and M_j be the actual finish time and the scheduled mode of job j, respectively. X_j and M_j are both unspecified unless job j is assigned a scheduled completion time of t (X_j=t) and a scheduled mode of m (M_j=m).

(5) The enumeration process begins by assigning job 1, mode 1 at its earliest possible completion time to the schedule. First, check the feasibility of nonrenewable resources by using the nonrenewable resource bound which is the sum of the cumulative total resource for all jobs before job j, the total resource for job j under the current mode, and the cumulative minimum total resource for all jobs beyond job j until the last job. If the

nonrenewable resource bound is less than or equal to the nonrenewable resource limit, check the feasibility of renewable resources. Otherwise, an attempt is made to assign the next mode at its earliest possible completion time. If none of the modes of the job are feasible, begin the backtracking process.

(6) The feasibility checking process for renewable resources depends on the type of the job. For a splittable job, the resource availabilities of all renewable resource types are tested each time the job is assigned an active time. If the job can be assigned at the active time, reduce the array of renewable resource availability and find the next active time until the total number of active periods is equal to the job duration. A completion time is then assigned for the job. Otherwise, increase time by one unit and try to find the earliest possible active time within its range by using one of its available modes. For a nonsplittable job, find the earliest contiguous interval that has a length equal to the job duration and that satisfies the current availability of all renewable resource types. For example, if job j mode m with a duration of d_{jm} is assigned a completion time of t, then the

contiguous interval that has a length of d_{jm} periods is searched and checked for the renewable resource feasibility, i.e., check whether renewable resource type k consumed by job j, mode m is less than or equal to the current availability of renewable resource type k between time period $(t-d_{jm}+1)$ and t. If at least one type of renewable resource does not satisfy the above condition, increase the completion time by 1 and find the earliest possible completion time. For both splittable and nonsplittable jobs, if the current mode of the job cannot be assigned, try to assign the next mode at its earliest possible completion time. If none of the modes of the job is feasible, then the backtracking process begins.

(7) The same enumeration process is applied to all jobs with an attempt to assign the first mode of each job at its earliest possible completion time to the schedule. Jobs must be selected for assignment in ascending order of the job labeling numbers to ensure that the precedence relationships between jobs are maintained. The earliest start time of each job must be computed based on the current scheduled time of all predecessor jobs. After the job is assigned a completion time, the resource availability array must be reduced in accordance

with the amount of resources consumed by the scheduled mode of the job. For a splittable job, only the nonrenewable resource array of availability is reduced at this step because the array of renewable resource availability has already been reduced when the active time is found.

(8) In the backtracking process, if job j cannot be assigned to the schedule by using any mode, a job that was previously assigned before job j must be selected for rescheduling according to one of the backtracking rules given in Table 5.2. Let job s be the selected job for rescheduling; job s was previously assigned by using mode m and was scheduled to complete at time t. The resource availability arrays of all jobs from job s to job j-1 are increased based on their scheduled times. Find the earliest possible completion time for job s using the same process. If job s can be completed within its range, reduce the array of resource availability by the amount consumed by job s and begin the augmentation process at job s+1, mode 1. If none of the modes of job s is feasible, reduce the array of resource availability for all jobs from job s to job j-1 by the amount previously increased to obtain the original partial schedule. Select a new job for rescheduling from

the same partial schedule by using the backtracking rule previously used.

(9) The augmentation and backtracking processes continue systematically until one of the following conditions is satisfied:

(a) All jobs are assigned a completion time, and an improved project completion time is found.

(b) The backtracking process is made past job 1, and the current incumbent solution is optimal.

(10) If an improved project completion time is found, set a new incumbent solution (T^*) equal to the completion time of the last job in the complete schedule. If T^* exceeds the best known lower bound such as the critical path early completion time, then the late finish times L_j of each job are tightened by the quantity $[L_N-(T^*-1)]$. A comparison is made between the tightened latest finish time and the completion time of each job from the incumbent complete schedule to determine a new unassigned job. After the unassigned job is identified, begin the augmentation process on the unassigned job using the next mode (mode after the currently scheduled mode).

(11) Optimality is reached when either the backtracking process is made past job 1, or an improved project

completion time is found to be equal to a proven theoretical bound such as the critical path early completion time.

<u>Example</u>: After the labeling process is performed, set an incumbent project completion time (T^*) equal to 16 (=4+4+3+5) which is the sum of maximum job duration of all jobs. Based on T^* (16), the latest finish times of jobs 1, 2, 3, and 4 become 9, 12, 12, and 16, respectively. Begin the enumeration process at job 1, mode 1. Check the feasibility of the nonrenewable resource (total cost) at the earliest possible completion time of job 1, mode 1 (period 3). Compare the nonrenewable resource bound of job 1, mode 1 (30+20+10+40=100) with the nonrenewable resource limit (200). Since the value is less than the limit, the assignment is feasible. If this were not feasible (e.g., the nonrenewable resource bound of job 1 mode 1 is 300), then check for the feasibility of job 1, mode 2 at its earliest possible completion time (period 4). If none of the modes of job 1 were feasible, begin the backtracking process.

Since the nonrenewable resource is feasible for job 1, mode 1 at period 3, the renewable resource feasibility is tested. Compare the renewable resource consumption of job 1, mode 1 (5) with the renewable resource limit (7) at periods 1, 2, and 3. Since the value is less than the limit, job 1 is assigned a scheduled completion time of 3 and a scheduled mode of 1. The resource availability arrays of both renewable and nonrenewable resources are decreased by the amount consumed by job 1, mode 1, e.g., the array of nonrenewable resource is reduced to 170 (=200-30), and the array of renewable resource is reduced to 2 (=7-5). The enumeration process then begin on the next job (job 2). If this were not feasible, increase the completion time by 1 and find the earliest possible completion time. If job 1, mode 1 could not be assigned by its latest completion time (period 9), then check for the feasibility of job 1, mode 2 at its earliest possible completion time. If none of the modes of job 1 were feasible within its latest completion time (period 9), begin the backtracking process.

The process continues systematically until all jobs are assigned to the schedule. A complete schedule is found as follows: job 1 is assigned a scheduled completion time of 3 and a scheduled mode of 1, job 2 is assigned a scheduled completion time of 6 and a scheduled mode of 1, job 3 is assigned a scheduled completion time of 8 and a scheduled mode of 1, and job 4 is assigned a scheduled completion time of 12 and a scheduled mode of 1.

Since the project completion time of the complete schedule (12) is less than the incumbent value (16), an improved solution is found, and the complete schedule becomes an incumbent solution. Reduce the latest finish times of all jobs by 5 (=16-12+1); thus, the tightened latest finish of jobs 1, 2, 3, and 4 become 4, 7, 7, and 11, respectively. The completion times of job 3 and job 4 in the incumbent solution are larger than their tightened latest finish times, but job

3 has a lower order number than job 4; thus, job 3 is an unassigned job. Restart the enumeration process on job 3.

From the example job 3 cannot be assigned to the schedule by using any mode within its latest completion time (7). Thus, the backtracking process begin. Let assume that the LIFO backtracking rule is used, job 2 is then selected for rescheduling. Increase the resource availability array at the scheduled times of job 2 (period 4, 5 and 6) by the amount consumed by job 2, mode 1 (5 units for renewable resource, and $20 for nonrenewable resource). Increase the completion time of job 2, mode 1 by 1 and find the earliest possible completion time. The process continues systematically until all jobs are assigned to the schedule. A complete schedule is found as follows: job 1 is assigned a scheduled completion time of 3 and a scheduled mode of 1, job 2 is assigned a scheduled completion time of 7 and a scheduled mode of 2, job 3 is assigned a scheduled completion time of 6 and a scheduled mode of 2, and job 4 is assigned a scheduled completion time of 11 and a scheduled mode of 1.

Since the project completion time of the complete schedule (11) is less than the incumbent value (12), an improved solution is found, and the complete schedule becomes an incumbent solution. Reduce the latest finish times of all jobs by 1 (=11-11+1); thus, the tightened latest finish of jobs 1, 2, 3, and 4 become 3, 6, 6, and 10, respectively. The completion times of job 2 and job 4 in the incumbent solution are larger than their tightened latest finish times, but job 2 has a lower order number than job 4; thus, job 2 is an unassigned job. Restart the enumeration process on job 2, and the process continues as before.

No further improvement has been found, and the backtracking process is made past job 1. Thus, the current incumbent solution is an optimal solution with the optimal project completion time of 11.

5.3.2 Cost Minimization Algorithm

The solution procedure for cost minimization problems is described in the following steps:

(1) Perform the labeling process.

(2) Set an incumbent project total cost (C^*) equal to a known heuristic value. If a heuristic value is not known, set C^* equal to the sum of maximum total cost of all jobs.

(3) Set a project completion time equal to the desired project completion time (g). Calculate the latest finish time of each job by using the shortest duration mode and the desired project completion time. The range will remain the same throughout the process since the objective is to find the minimum cost schedule within the desired project completion time.

(4) is same as Step (4) for the time minimization

objective (section 5.3.1).

(5) The enumeration process begins by assigning job 1, mode 1 at its earliest possible completion time to the schedule. Before evaluating the resource feasibility, check the feasibility of total project cost by using the total cost bound which is the sum of the cumulative total cost for all jobs before job j, the total cost for job j under the current mode, and the cumulative minimum total cost for all jobs beyond job j until the last job. If the total cost bound is less than or equal to the project total cost, check for resource feasibility. Otherwise, begin the backtracking process. If job j, mode m cannot be assigned to the schedule due to the above condition, any mode of job j after mode m is also infeasible because modes are ranked in increasing order of total cost in the cost minimization objective. The resource feasibility test begins by checking the feasibility of nonrenewable resources through the nonrenewable resource bound. If the nonrenewable resource bound is less than or equal to the nonrenewable resource limit, check the feasibility of renewable resources. Otherwise, an attempt is made to assign the next mode at its earliest possible completion time. If none of the modes of the job are feasible,

begin the backtracking process.

(6) - (9) are same as Steps (6) - (9) for the time minimization objective (section 5.3.1).

(10) If an improved project total cost is found, set a new incumbent solution (C^*) equal to the cumulative total cost of the last job. If C^* exceeds the best known lower bound such as the sum of minimum total cost of all jobs, then set the tightened project total cost equal to (C^*-e) where e is an amount of cost reduction which depends on desired accuracy. A comparison is made between the tightened project total cost and the cost bound of each job based on the incumbent complete schedule to determine a new unassigned job. After the unassigned job is specified, begin the backtracking process. The other modes of the unassigned job can be eliminated from consideration because modes are ranked in increasing order of total cost.

(11) Optimality is reached when either the backtracking process is made past job 1, or an improved project total cost is found to be equal to a proven theoretical bound such as the sum of minimum total cost of all jobs.

Example: After the labeling process is performed, set an incumbent project total cost (C^*) equal to \$160 (=40+40+30+50) which is the sum of maximum total cost of all jobs. Let assume that the project is desired to be completed within period 11. The project completion time is set to 11, and the latest finish times of jobs 1, 2, 3, and 4 are 4, 7, 7, and 11, respectively. First check the feasibility of total project cost for job 1, mode 1 by comparing the total cost bound for job 1, mode 1 (100=30+20+10+40) to the incumbent project total cost (160). Since the total cost bound is less than the incumbent project total cost, the assignment is feasible. The enumeration process is then performed as explained in the time minimization objective. If this were not feasible (e.g., the total cost bound is 200), then begin the backtracking process.

The same complete schedule is obtained as in the time minimization objective. Since the project total cost of the complete schedule is 140 (=30+40+30+40) which is less than the incumbent value (160), an improved solution is found, and the complete schedule is a new incumbent solution. Reduce the project total cost by 1, the tightened project total cost becomes 139 (=140-1). Compare the tightened project total cost with the cost bound of jobs 1, 2, 3, and 4 which are 100 (=30+20+10+40), 120 (=30+40+10+40), 140 (=30+40+30+40), 140 (=30+40+30+40), respectively. The cost bounds of job 3 and job 4 are larger than the tightened project total cost, but job 3 has a lower order number than job 4; thus, job 3 is an unassigned job. Restart the enumeration process on job 3, and the process continues as before.

No further improvement is found, and the backtracking process is made past job 1. Thus, the current incumbent solution is an optimal solution with the optimal project total cost of 140.

5.3.3 Resource Leveling Algorithm

The solution procedure for resource leveling problems is described in the following steps:

(1) Perform the labeling process.

(2) Set an incumbent objective value (D^*) equal to a known heuristic value. If a heuristic value is not known, set D^* equal to

$$\sum_{\forall t} |\text{desired rate - resource used}|$$

when all jobs are assigned sequentially.

(3) Set a project completion time equal to the desired project completion time (g). Calculate the latest finish time of each job by using the shortest duration mode and the desired project completion time. The range will remain the same throughout the process since the objective is to minimize resource deviation within the desired completion time.

(4) - (8) are same as Steps (4) - (8) for time minimization objective (section 5.3.1).

(9) After a completion time is found for the last job, the total resource deviation is computed and compared to the current objective value. If the total resource deviation exceeds the current

objective value, an attempt is made to find the earliest possible completion time that satisfies resource feasibility and has total resource deviation less than or equal to the current objective value. If none of the modes of the last job is feasible within its range, then the backtracking process begins.

(10) The augmentation and backtracking processes continue systematically until one of the following occurs.

(a) All jobs are assigned a completion time, and an improved objective value is found.

(b) The backtracking process is made past job 1, and the current incumbent solution is optimal.

(11) If an improved total resource deviation is found, set a new incumbent objective value (D^*) equal to the total resource deviation. If D^* exceeds the best known lower bound such as a zero resource deviation, then set the tightened objective value equal to (D^*-f) where f is a reduction in resource deviation which depends on a desired accuracy. Since the last job is an unassigned job, begin the augmentation process on the last job by using the scheduled mode in the incumbent solution. This will check the feasibility of the last job at a finish time after the scheduled finish time in the

incumbent solution.

(12) Optimality is reached when either the backtracking process is made past job 1, or an improved objective value is found to be equal to a proven theoretical bound such as a zero deviation.

Example: After the labeling process is performed, set an incumbent objective value (D^*) equal to 65 (=16+20+9+20) which is the sum of maximum resource deviation of all jobs. Let assume that the project is desired to be completed within period 11. The project completion time is set to 11, and the latest finish times of jobs 1, 2, 3, and 4 are 4, 7, 7, and 11, respectively. The enumeration process is performed as explained in the time minimization objective.

The same complete schedule is obtained as in the time minimization objective. The total resource deviation is computed after job 4 is assigned. Since the total resource deviation of the complete schedule (26) is less than the incumbent value (65), an improved solution is found, and the complete schedule is a new incumbent solution. Reduce the total resource deviation by 1, and the tightened objective value becomes 25 (=26-1). Job 4 is an unassigned job, and restart the enumeration process on job 4. The process continues systematically as before.

No further improvement is found, and the backtracking process is made past job 1. Thus, the current incumbent solution is an optimal solution with the optimal total resource deviation of 26.

5.3.4 Multiple Objective Algorithm

The multiple objective algorithm considers three objectives with different levels of priority. The algorithm allows the user to specify priority level for each goal. However, for the purpose of illustration it is assumed that the highest priority goal is to minimize time beyond the desired due date (tardiness), the second highest priority goal is to minimize cost beyond the desired project total cost, and the lowest priority goal is to minimize total resource deviation from some desired limit.

Let

T^* = project completion time of the incumbent solution

C^* = project total cost of the incumbent solution

D^* = total resource deviation of the incumbent solution

g = desired project completion time (due date)

c = desired project total cost

and $P1 >>> P2 >>> P3$ where P1, P2 and P3 are the priorities (weights) for goals 1, 2, and 3, respectively.

The mathematical formulation for the goal programming is as follows:

Goal 1: Minimize time beyond the desired due date

Min f^+

where $T^*+f^--f^+ = g$

Goal 2: Minimize cost beyond the desired project total cost

Min t^+

where $C^*+t^--t^+ = c$

Goal 3: Minimize total resource deviation from the desired limits

$$\text{Min} \sum_{\forall k} v_k^-$$

where $(\text{total resource used})_k + v_k^-$

$= (\text{desired limit})_k$

The algorithm for multiple objective goal programming problems is as follows:

GOAL 1: Minimize time beyond the desired due date.

(1) Perform the labeling process.

(2) If the desired due date is known, set T^* equal to the desired due date. Otherwise, set T^* equal to the sum of all maximum job durations. Set the project total cost and the total resource deviation equal to their maximum values.

(3) Calculate the latest finish time of each job by using the shortest duration mode and the project completion time of T^*.

(4) is same as Step (4) for the time minimization objective (section 5.3.1).

(5) is same as Step (5) for the cost minimization objective (section 5.3.2).

(6) - (8) are same as Steps (6) - (8) for the time minimization objective (section 5.3.1).

(9) is same as Step (9) for the resource leveling objective (section 5.3.3).

(10) The augmentation and backtracking processes continue systematically until one of the followings conditions is satisfied:

(a) All jobs are assigned a completion time: an improved objective value is found.

(b) The backtracking process is made past job 1: the current incumbent solution is optimal.

(11) After an improved project completion time is found, the following process is performed.

Case 1: The desired due date is specified.

If the improved project completion time is found to be less than or equal to the desired due date, stop the process for Goal 1 and start Goal 2. Otherwise, increase the desired due date and the latest finish of each job by 1 and begin the augmentation process at job 1 mode 1. The process continues systematically until the above condition is met.

Case 2: The desired due date is not specified.

If the improved project completion time is found, set a new incumbent solution (T^*) equal to the completion time of the last job in the complete schedule. If T^* exceeds the best known lower bound such as the critical path early completion time, then the late finish times L_j of each job are tightened by the quantity $[L_N-(T^*-1)]$. A comparison is made between the tightened latest finish time and the completion time of each job from the incumbent complete schedule to determine a new unassigned job. After the unassigned job is identified, begin the augmentation process on the unassigned job using the next mode (the mode after the currently scheduled mode).

(12) Optimality is reached when either the backtracking process is made past job 1, or an improved project completion time is found to be equal to a proven theoretical bound such as the critical path early completion time.

<u>Example</u>: Let assume that the desired due date is 11. Set the incumbent project completion time (T^*) to 11, the project total cost to 160, and the total resource deviation to 65. Based on T^*, the latest finish times of jobs 1, 2, 3, and 4 are 4, 7, 7, and 11, respectively. First check the feasibility of total project cost for job 1, mode 1 by comparing the total cost bound for job 1, mode 1 (100=30+20+10+40) to the

incumbent project total cost (160). Since the total cost bound is less than the incumbent project total cost, the assignment is feasible. The enumeration process is then performed as explained in the time minimization objective. If this were not feasible (e.g., the total cost bound is 200), then begin the backtracking process.

The same complete schedule is obtained as in time minimization objective. The total resource deviation is computed after job 4 is assigned. Since the total resource deviation of the complete schedule (26) is less than the incumbent value (65), the complete schedule is feasible.

Since the project completion time of the complete schedule (11) is equal to the desired due date, stop the process for Goal 1 and proceed to Goal 2. If this were not feasible, increase the due date by 1, and restart the process at job 1, mode 1. The process continues systematically until the project completion time of a complete schedule is less than or equal to the desired due date which has been increased.

GOAL 2: Minimize cost beyond the desired project total cost.

(1) Perform the labeling process.

(2) If the desired project total cost is specified, set C^* equal to the desired project total cost. Otherwise, set C^* equal to the sum of maximum total cost of all jobs. Set the project completion time

equal to the desired due date if the value obtained from Goal 1 is less than or equal to the desired due date. Set the project completion time equal to the value obtained from Goal 1 if the value obtained from Goal 1 is larger than the desired due date. Set the total resource deviation equal to its maximum value.

(3) Calculate the latest finish time of each job by using the shortest duration mode. The value of the latest finish of each job will remain unchanged throughout the process.

(4) - (10) are same as Steps (4) - (10) in Goal 1.

(11) After an improved project total cost is found, the following process is performed.

Case 1: The desired project total cost is specified.

If the improved project total cost is found to be less than or equal to the desired project total cost, stop the process for Goal 2 and start Goal 3. Otherwise, increase the desired project total cost by 1 and begin the augmentation process at job 1 mode 1. The process continues systematically until the above condition is met.

Case 2: The desired project total cost is not specified.

If an improved project total cost is found, set a new incumbent solution (C^*) equal to the cumulative total cost of the last job. If C^* exceeds the best known lower bound such as the sum of minimum total cost of all jobs, then set the tightened project total cost equal to (C^*-e) where e is an amount of cost reduction which depends on a desired accuracy. A comparison is made between the tightened project total cost and the cost bound of each job based on the incumbent complete schedule to determine a new unassigned job. After the unassigned job is specified, begin the backtracking process. The other modes of the unassigned job can be eliminated from consideration because modes are ranked in increasing order of total cost.

(12) Optimality is reached when either the backtracking process is made past job 1, or an improved project total cost is found to be equal to a proven theoretical bound such as the sum of minimum total cost of all jobs.

<u>Example</u>: Let assume that the desired project total cost is 150. Set the incumbent project total cost (C^*) to 150, the project completion time to 11, and the total resource deviation to 65. Based on the project completion time of 11,

the latest finish times of jobs 1, 2, 3, and 4 become 4, 7, 7, and 11, respectively. The same process is then performed as explained in Goal 1.

Since the project total cost of the complete schedule (140) is less than the desired project total cost, stop the process for Goal 2 and proceed to Goal 3. If this were not feasible, increase the desired project total cost by 1, and restart the process at job 1, mode 1. The process continues systematically until the project total cost of a complete schedule is less than or equal to the desired level which has been increased.

GOAL 3: Minimize total resource deviation from desired limit.

(1) Perform the labeling process.

(2) If the desired level of total resource deviation is specified, set D^* equal to the desired level of total resource deviation. Otherwise, set D^* equal to $\sum_{\forall t}$ |desired rate - resource used| when all jobs are assigned sequentially. Set the project completion time equal to the desired due date if the value obtained from Goal 1 is less than or equal to the desired due date. Set the project completion time equal to the value obtained from Goal 1 if the value obtained from Goal 1 is larger than the desired due date. Set the project total cost equal to the desired project total cost if the

value obtained from Goal 2 is less than or equal to the desired project total cost. Set the project total cost equal to the value obtained from Goal 2 if the value obtained from Goal 2 is larger than the desired project total cost.

(3) Calculate the latest finish time of each job by using the shortest duration mode. The value of the latest finish of each job will remain unchanged throughout the process.

(4) - (10) are same as steps (4) - (10) in Goal 1.

(11) After an improved total resource deviation is found, the following process is performed.

Case 1: The desired level of total resource deviation is specified.

If the improved total resource deviation is found to be less than or equal to the desired level of total resource deviation, the process ends. Otherwise, increase the desired level of total resource deviation by 1 and begin the augmentation process at job 1 mode 1. The process continues systematically until the above condition is met.

Case 2: The desired level of total resource deviation is not specified.

If the improved total resource deviation is found, set a new incumbent objective value (D^*)

equal to the total resource deviation. If D^* exceeds the best known lower bound such as a zero resource deviation, then set the tightened objective value equal to (D^*-f) where f is a reduction in resource deviation which depends on a desired accuracy. The last job is an unassigned job when an improved solution is found. Thus, begin the augmentation process on the last job by using the scheduled mode in the incumbent solution. This will check the feasibility of the last job at finish time after the scheduled finish time in the incumbent solution.

(12) Optimality is reached and the process ends when either the backtracking process is made past job 1, or an improved objective value is found to be equal to a proven theoretical bound such as a zero deviation.

Example: Let assume that the desired total resource deviation is 40. Set the incumbent total resource deviation (D^*) to 40, the project completion time to 11, and the project total cost to 150. Based on the project completion time of 11, the latest finish times of jobs 1, 2, 3, and 4 become 4, 7, 7, and 11, respectively. The same process is then performed as explained in Goal 1.

Since the total resource deviation of the complete schedule (26) is less than the desired level, stop the process. If this were not feasible, increase the desired total resource deviation by 1, and restart the process at job 1, mode 1. The process continues systematically until the total resource deviation of a complete schedule is less than or equal to the desired level which has been increased.

5.4 Implementation of the Algorithms

A computer program was written for each algorithm using the C programming language. The computer code can reliably handle up to 30 jobs, 3 modes of operation, 1 type of nonrenewable resource, 6 types of renewable resources, and 200 time periods. The code is general enough to handle larger problems; however, it has not been tested for problems exceeding the characteristics mentioned earlier. The code is compiled by using MS QUICKC compiler (version 2.50) with a large memory model. The programs were run on IBM compatible machines with 33 MHz Intel 486DX processor in the IME computer lab at Oregon State University.

CHAPTER 6

TESTING OF THE ALGORITHMS

6.1 Test Environments

Three types of problems normally used for testing algorithms are computer-generated problems, problems taken from published literature, and case studies adapted from practice. The computer-generated problems are used in this study for testing the algorithms. The system is then illustrated using a case study.

Very few problems in reported literature include all the network characteristics investigated in this study; these problems are therefore of little use in evaluating this study. Actual problems from practice are not used because to identify an appropriate implementation environment, to obtain results using the algorithms, and to compare these results with actual outcomes is a lengthy process, and beyond the time constraints for this study.

It was therefore decided to use computer-generated problems in evaluating the algorithms developed in this research, and then to use these results to identify implementation guidelines for practice. Furthermore, by using computer-generated problems, the area of interest can be thoroughly investigated while controlling the impact of other factors.

6.1.1 Computer-Generated Problems

The creation of computer-generated problems is based on the methods used by Davis (1968), Johnson (1967), and Talbot (1982). The computer-generated data includes: (1) splittable jobs, (2) number of modes and modes of operation for each job, (3) job durations, (4) precedence relationships, and (5) the amount of renewable resources consumed by each mode of each job. Details of the procedures are discussed below; Appendix 1 explains how the probability distributions are obtained.

1. Project Specifications

To obtain computer-generated data, the following input parameters must be specified:

(1) number of jobs (n)

(2) number of splittable jobs (s)

(3) maximum number of modes of operation (m)

(4) number of renewable resource types (r)

(5) probability distribution of modes of operation, of job durations and of renewable resources consumption

(6) network configuration such as number of tier levels and number of jobs in each tier.

2. Assignment of Splittable Jobs

(1) Generate s random numbers between 1 and n, where s represents the number of splittable jobs to be generated, and n is the total number of jobs.

(2) Splittable jobs are the jobs that have their job

numbers same as the generated random numbers.

3. Assignment of Modes of Operation

3.1) Determine the number of modes for each job

(1) Generate n random numbers between 1 and m, where n is the number of jobs, and m is the maximum number of modes.

(2) Assign number of modes to each job according to the generated random numbers.

3.2) Determine modes of operation for each job

(1) Generate successive random numbers between 0 and 1 for each job.

(2) Assign corresponding modes to each job based on the number of modes obtained from (3.1) and the following distribution (assuming that m is equal to 3).

If a job has only 1 mode,

MODE	1	2	3
PROBABILITY	.573	.298	.129

If a job has 2 modes,

MODES	1,2	1,3	2,3
PROBABILITY	.436	.351	.213

If a job has 3 modes, assign all three modes (1, 2 and 3) to the job.

4. Random Assignment of Job Durations

(1) Generate successive random numbers between 0 and 1 for each mode of each job.

(2) Assign a duration to each mode of each job based on its corresponding mode (assuming that m is equal to 3) and

the following distribution. This assumes that jobs using mode 1 may have durations of 1, 2, or 3 time units; jobs using mode 2 may have durations of 4, 5, or 6 time units; and jobs using mode 3 may have durations of 7, 8, or 9 time units.

MODE 1:	DURATION	1	2	3
	PROBABILITY	.272	.403	.325
MODE 2:	DURATION	4	5	6
	PROBABILITY	.409	.326	.265
MODE 3:	DURATION	7	8	9
	PROBABILITY	.465	.318	.217

5. Creation of Precedence Relationships

Precedence relationships are randomly assigned within a specified range, not from all possibilities of job combinations. This prevents the creation of redundant relationships, and gives a higher degree of control over network configuration with some degree of randomness. The procedure is as follows:

(a) Create a (n x n) precedence matrix.

(b) Assign 2 to the cells where job j does not precede job k but the job orders of j and k can be switched, i.e., job j can be performed either before or after job k. Thus, assign 2 to all (j,k) cells having the same tier number.

(c) Assign 3 to the cells where job j does not precede job k and the job orders of j and k cannot be switched, i.e., job j must be performed before job k when $j < k$, and job k must be performed before job j when $j > k$. Thus, assign 3 to the following cells:

(i) all cells where j is equal to k

(ii) all (j,k) cells where tier number of job j (t_j) is greater than tier number of job k (t_k)

(iii) all (j,k) cells where (t_j+1) is less than t_k.

(d) Assign 1 to the cells where job j precedes successor job k by using the following procedures:

(i) In each row, a job j can select a particular successor job k with some probability.

(ii) If a job j is not assigned a successor from the previous step, randomly select one successor job from a set of candidates with the same probability for each candidate.

(iii) In each column, each job k that is not assigned a predecessor from the previous step can select a predecessor job j from a set of candidates with the same probability for each candidate.

(e) Assign 3 to any (j,k) cell that is still unassigned.

6. <u>Assignment of Renewable Resources Consumption</u>

(a) Generate successively a random number between 0 and 1 for each renewable resource type of each job.

(b) For each type of renewable resources, the amount of resource consumption by each job is determined by the following distribution:

UNITS OF CONSUMPTION	1	2	3	4	5	6
PROBABILITY	.208	.236	.213	.172	.117	.054

(c) Since dummy resources are used to indicate the difference between modes of operation, each mode is assigned 1 unit of its corresponding dummy resource and 0 unit of the other dummy resources.

To show how resources are assigned to each mode of each job, consider an example in Table 6.1. From the example, resource types 1, and 2 are dummy resources, and resource types 3, and 4 are renewable resources. The amount of resource types 3, and 4 consumed by job 1 are the same for both modes 1, and 2, i.e., 2 units of resource type 3, and 3 units of resource type 4. The two modes of job 1 are differentiated by assigning 1 unit of dummy resource type 1, and 0 unit of dummy resource type 2 to mode 1; and 0 unit of dummy resource type 1, and 1 unit of dummy resource type 2 to mode 2.

TABLE 6.1 AN EXAMPLE OF RESOURCE ASSIGNMENT

JOB	MODE	RESOURCE TYPE			
		1	2	3	4
1	1	1	0	2	3
1	2	0	1	2	3

6.1.2 Network Characteristics of Computer Problems

Davis (1975) introduced the network summary measures which can be used to specify the network characteristics for resource-constrained project scheduling problems. These fall into three general classes: (1) network size, shape, and logic, (2) time characteristics, and (3) resource characteristics. However, only some of the measures are used in this study due to the difference in problem characteristics. Since the problem investigated by Talbot (1982) is the same as the one in this study, the network measures reported in Talbot's paper are used as guidelines for this study.

The network measures reported for the problems with time minimization objective include:

(1) Network complexity (C) is defined as the ratio of the number of arcs (precedence relationships) to the number of jobs.

(2) Critical path early finish time obtained when there is no resource restriction.

(3) Optimal project completion time obtained under a specified resource limitation.

In addition to the above network measures, internal complexity (C^*) is introduced to measure the relative complexity of a network, and is defined as the ratio of the number of arcs to the maximum number of arcs. The internal complexity (C^*) is different from the network complexity (C) in that C^* measures the relative complexity within the same network, while C measures the complexity across networks. In a computer-generated network, the minimum and maximum numbers of arcs can be determined as follows:

Let

i = tier level, and

n = maximum number of tier levels.

Then the minimum number of arcs is given by

$$\sum_{i=1}^{n-1} MAX\ (number\ of\ jobs\ at\ tier\ i,\ number\ of\ jobs\ at\ tier\ i{+}1)\ ,$$

and the maximum number of arcs is given by

$$\sum_{i=1}^{n-1} (number\ of\ jobs\ at\ tier\ i) * (number\ of\ jobs\ at\ tier\ i{+}1)$$

Since the minimum and maximum numbers of arcs can be determined for a computer-generated problem, different levels of complexity can be set for a problem based on the values specified for internal complexity.

Eighteen test problems are generated with different number of jobs, network shapes, and complexity based on network configurations and level of complexity given in Table 6.2. Details of the test problems including the sample problem from Talbot's paper (1982) are given in Appendix 2.

Five different problem sets were originally generated for testing the algorithms. Details of each problem set is given in Appendix 3, and is summarized as follows:

TABLE 6.2 SUMMARY OF NETWORK CONFIGURATIONS

# of jobs	Shape #	Shape	Level of Complexity	Value of C^*
10	1	x x x x x x x x x x	Low Medium High	.55 - .70 .71 - .85 .86 - 1
10	2	x x x x x x x x x x x	Low Medium High	.50 - .67 .68 - .83 .84 - 1
20	3	x x x x x x x x x x x x x x x x x x x x	Low Medium High	.32 - .54 .55 - .77 .78 - 1
20	4	x x x x x x x x x x x x x x x x x x x x	Low Medium High	.29 - .51 .52 - .75 .76 - 1
30	5	x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x	Low Medium High	.24 - .48 .49 - .74 .75 - 1
30	6	x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x	Low Medium High	.22 - .47 .48 - .73 .74 - 1

1. D1 consists of 18 computer-generated problems used for the time minimization objective. The test is performed on 8 job labeling rules given in Table 5.1, and on 12 backtracking rules including LIFO, MILF, MALF, MITF, MATF, MID, MAD, MIRD, MARD, MITR, MATR, and RAN (for definitions, see Table 5.2).
2. D2 consists of 18 computer-generated problems and the sample problem from Talbot's paper. It is used for the time minimization objective, and also contains the nonrenewable resource constraint. The test is performed on 4 selected job labeling rules (MAAD, MAAR, MILAD, and RAN), and on the 12 backtracking rules stated in (1) and the algorithm developed by Talbot (1982).
3. D3 consists of 18 computer-generated problems used for the cost minimization objective. The test is performed on 4 selected job labeling rules (MAAD, MAAR, MILAD, and RAN), and on all 14 backtracking rules given in Table 5.2.
4. D4 consists of 9 computer-generated problems selected from D3 and the sample problem from Talbot's paper. It is used for the cost minimization objective. The test is performed on 4 selected job labeling rules (MAAD, MAAR, MILAD, and RAN), and on all 14 backtracking rules given in Table 5.2 and the algorithm developed by Talbot

(1982).

5. D5 consists of 18 computer-generated problems used for the resource leveling objective. The test is performed on 4 selected job labeling rules (MAAD, MAAR, MILAD, and RAN), and on 6 backtracking rules (LIFO, MIRD, MARD, MITR, MATR, and RAN).

6.2 Design of Experiments

The experiment is designed such that each tested problem is considered as a homogeneous block, and different combinations of job labeling rules and backtracking rules are tested for each problem. The results obtained from each run is the computation time required to reach optimality. If the optimality cannot be reached within a given time limit, the related objective value is observed instead. Since the statistical model includes two factors, job labeling rules and backtracking rules, the two-factor factorial in randomized complete block is used as the statistical model in this study. The model can be written as (Ostle and Malone 1988, pg.392)

$$Y_{ijk} = \mu + \rho_i + \alpha_j + \beta_k + (\alpha\beta)_{jk} + \varepsilon_{ijk} \qquad (6.1)$$

where $i = 1, \ldots, r$

$j = 1, \ldots, a$

$k = 1, \ldots, b$

r = number of blocks (problems)

a = number of levels of factor A (job labeling rules)

b = number of levels of factor B (backtracking rules)

μ = true mean effect

ρ_i = true effect of the i^{th} block

α = effect of factor A

β = effect of factor B

$\alpha\beta$ = interaction of factor A and B

$\varepsilon_{ijk} \sim NID(0, \sigma^2)$

and $e_{ijk} = Y_{ijk} - \overline{Y}_{.jk} - \overline{Y}_{i..} + \overline{Y}_{...}$

The data sets used in statistical analysis are summarized in Table 6.3. These are based on the five problem sets (D1 through D5) defined in Section 6.1.2. The following tests were performed on these data sets as explained below, and summarized in Table 6.4:

1. TEST1 uses DATA1 to test the differences among job labeling rules. It is used as a preliminary test to eliminate some job labeling rules. Duncan's multiple-range test is used to compare all pairs of the job labeling rules.
2. TEST2 uses DATA2 to test the differences among the 12 backtracking rules. Duncan's multiple-range test is used to compare all pairs of the backtracking rules. Dunnett's t test is used to compare all backtracking rules with the control rule (LIFO).
3. TEST3 uses DATA3 to test the difference between the 12 backtracking rules and the control treatment

TABLE 6.3 SUMMARY OF DATA SETS USED IN STATISTICAL ANALYSIS

Data Set	Objective Function	# of problems tested	# of job labeling rules	# of backtracking rules	Comments
DATA1	time	18	8	12	same as D1
DATA2	time	37	4	12	combine D1 with D2 on the 4 selected job labeling rules
DATA3	time	19	4	12	same as D2
DATA4	cost	18	4	14	rank each data in D3 based on its project total cost*
DATA5	cost	10	4	14	same as D4
DATA6	resource leveling	18	4	6	rank each data in D5 based on its total resource deviation*

* use the mid-rank approach to break ties

TABLE 6.4 SUMMARY OF STATISTICAL TESTS

Test#	Data Set	Objective Function	What to Compare?
TEST1	DATA1	time	difference among 8 job labeling rules
TEST2	DATA2	time	difference among 12 backtracking rules, and comparison to the control rule (LIFO)
TEST3	DATA3	time	compare the 12 backtracking rules to the control treatment (TALBOT)
TEST4	DATA4	cost	difference among the 14 backtracking rules, and comparison to the control rule (LIFO)
TEST5	DATA5	cost	difference among the 14 backtracking rules, comparison to the control rule (LIFO), and to the control treatment (TALBOT)
TEST6	DATA6	resource leveling	difference among the 6 backtracking rules, and comparison to the control rule (LIFO)

(TALBOT). Dunnett's t test is used to compare all backtracking rules with the control treatment (TALBOT).

4. TEST4 uses DATA4 to test the differences among the 14 backtracking rules based on the rank transformed data. Duncan's multiple-range test is used to compare all pairs of the backtracking rules. Dunnett's t test is used to compare all backtracking rules with the control rule (LIFO).

5. TEST5 uses DATA5 to test the differences among the 14 backtracking rules based on the real output data (computation time). Duncan's multiple-range test is used to compare all pairs of the backtracking rules. Dunnett's t test is used to compare all backtracking rules with the control rule (LIFO). In addition, the same data set is also used to test the difference between the 14 backtracking rules and the control treatment (TALBOT) using Dunnett's t test.

6. TEST6 uses DATA6 to test the differences among the 6 backtracking rules based on the rank transformed data. Duncan's multiple-range test is used to compare all pairs of the backtracking rules. Dunnett's t test is used to compare all backtracking rules with the control rule (LIFO).

All tests are performed at a significance level of .05.

The Duncan's multiple-range test is used because it is a powerful test which is very effective in detecting true differences in means. The Dunnett's t test is primarily used for making comparisons between each of the treatment means with the control; thus, it is used for testing the algorithms when comparisons with the control is desired. Several Dunnett's t tests have been performed including (1) Dunnett's two-tailed t test, testing if any treatment is significantly different from the control, (2) Dunnett's one-tailed t test, testing if any treatment is significantly smaller than the control, and (3) Dunnett's one-tailed t test, testing if any treatment is significantly larger than the control.

6.3 Model Adequacy Check

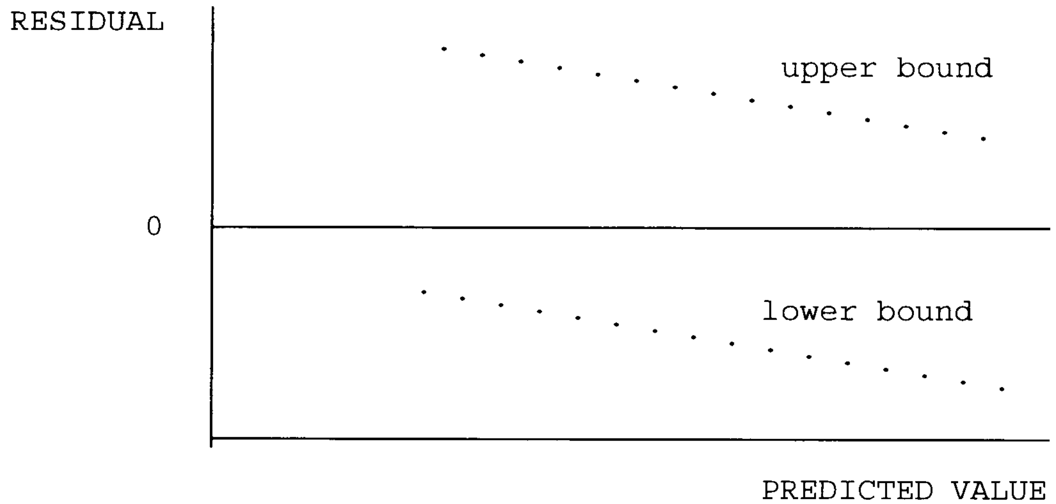
The validity of the statistical model given in equation (6.1) can be tested by using the residual plots and the normal probability plot. The residual plots include the plot between residuals and predicted values, the plot between residuals and blocks, and the plots between residuals and each of the two factors. The plots should show no relationship between the residuals and any variable. The assumptions of constant variances and normality of the errors are required for DATA1, DATA2, DATA3, and DATA5; but they are not required for DATA4 and DATA6 because the only assumption required for the rank transformed data is the random assignment of subjects to treatment and control. Departures from the assumptions of homogeneity of variances and normality usually causes slight effects on the analysis of the balanced fixed effects model such as the one used in this study.

The plots between residuals and predicted values of all data sets did not violate the constant variance and normality assumptions, except that some outliers exist in DATA2, DATA3 and DATA6. The outliers represent the problems which show extreme results on certain treatments. After the outliers have been removed, the plots are acceptable.

Certain patterns appear in the plots for DATA1, DATA2, DATA3, and DATA5 as shown in Figure 6.1. The upper bound line occurs because the computation time is controlled to be less than or equal to a predetermined time limit; and the lower bound line occurs because the computational time is controlled to be greater than or equal to zero. However, the patterns do not imply violation of the constant variance assumption because they are caused by the controlled condition of the experiments.

The plots between residuals and blocks (problems) show higher variability in some problems because some problems are more complex than others. The plots between residuals and each of the two factors are acceptable, particularly after some outliers have been removed from some tests. Again, the normal probability plots for all data sets are acceptable even though some data sets may have outliers. Since the above results show no signs of violating the model assumptions, the statistical

FIGURE 6.1 RESIDUAL PLOT



model given in equation (6.1) is valid and considered appropriate to be used in this study.

6.4 Test Results and Discussion

Since the experiments were performed on the network problems with 10, 20, and 30 jobs, the following conclusions should be applicable to most network problems up to 30 jobs. The focus of the experiments is to test the differences among the backtracking rules and compare the backtracking rules with the algorithm developed by Talbot. Only some job labeling rules are selected and used for testing the interaction between job labeling rules and backtracking rules. An extensive comparison between job labeling rules can be obtained from the research by Talbot (1982).

In the following figures, the rules are ordered in decreasing order of the mean values from left to right; thus, the rule on the extreme right has the lowest mean value and is better than the rule on the extreme left which has the highest mean value. The mean values represent either the average computation time required to reach optimal solutions or the average rank of objective function values. The underlines show that the rules above the line are not statistically different from each other, but are different from other values. Since the interactions between job labeling rules and backtracking rules was insignificant for all tests, so it is valid to make conclusions for the six tests.

TEST1 shows that there are some differences among the job labeling rules:

MIE $\underline{\text{MIL} \quad \text{RAN}}$ MAD MILAD $\underline{\text{MILSD} \quad \text{MAAR} \quad \text{MAAD}}$

MAAD, MAAR, MILAD, and RAN job labeling rules are selected for further analysis. MAAD is selected over MAD because both rules are based on the average duration criterion, but MAAD requires less computation time than MAD. MAAR is selected because it requires less computation time than other rules, and is based on the average resource demand criterion which is different from other rules. MILAD is selected over MIL and MILSD because these rules are based on the latest finish time criteria; but MILAD requires less computation time than MIL, and the computation time required by MILAD, unlike MILSD, differs from the time requires by MAAD and MAAR. RAN is selected because it is a standard rule which is applicable to most scheduling problems. MIE is not selected because it requires a significantly higher computation time than the other job labeling rules.

TEST2 shows that the only backtracking rule that is different from the others is MID.

MID MALF MAD MIRD MATF MATR MITR MILF MARD MITF LIFO RAN

However, if the outliers are removed, the test shows no difference among the backtracking rules:

MALF MAD MID MATR MATF MIRD MARD MILF MITF MITR LIFO RAN

Thus, it reveals no difference in the computation time required by each backtracking rule to reach optimal solutions in the time minimization problem.

TEST3 shows that all backtracking rules require significantly less computation time than the algorithm developed by TALBOT for the time minimization problem which includes the nonrenewable resource. Furthermore, the difference is significant both when the outliers are present and are removed from the data set.

TEST4 shows that there are some differences among the backtracking rules:

MILF MATF RAN MATC MITR MIRD MARD MID MATR MAD MITC MALF MITF LIFO

It also shows that in the cost minimization problem the objective function values obtained from MITC, MALF, and MITF backtracking rules are not significantly different from the value obtained from LIFO, while the other backtracking rules yield significantly higher objective function values.

<u>TEST5</u> shows that there are some differences among the backtracking rules:

LIFO MALF MILF MATC MATF RAN MARD MITF MID MIRD MATR MAD MITR MITC

It also shows that in the cost minimization problem the computation time required by MATR, MAD, MITR, and MITC backtracking rules are significantly less than the time required by LIFO, while it shows no difference between the other backtracking rules and LIFO.

Additional experiments show that all backtracking rules require significantly less computation time than the algorithm developed by TALBOT for the cost minimization problem.

<u>TEST6</u> shows that there are some differences among the backtracking rules:

RAN MARD MATR MIRD MITR LIFO

It also shows that in resource leveling problem the objective value obtained from LIFO is significantly better than the values obtained from other backtracking rules both when the outliers are present and when they are not. However, MITR also shows good performance in both the solutions and the computation time to reach the solutions.

From the experiments, it is found that the order of jobs can cause a large variation in computation time of the same problem. Using the same job labeling rule does not guarantee that the order of jobs will be the same because some job labeling rules may break ties arbitrarily. Thus, it is necessary to ensure that the same order of jobs is used when the experiments are performed on different backtracking rules or techniques. The order of modes and resources also affect the computation time, but to a lesser extent than the order of jobs, so the same orders must be used for all job labeling rules and backtracking rules. Other factors that might affect the computation time include the type of compiler and memory model used for the computer code, and the difference among computer machines.

Based on the experimental results of all tests, the following backtracking rules are selected for the application of the multiple objective algorithm to a warehouse project in the next section. The LIFO rule will be used as the backtracking rule for the objective of time minimization because there are no differences among the backtracking rules.

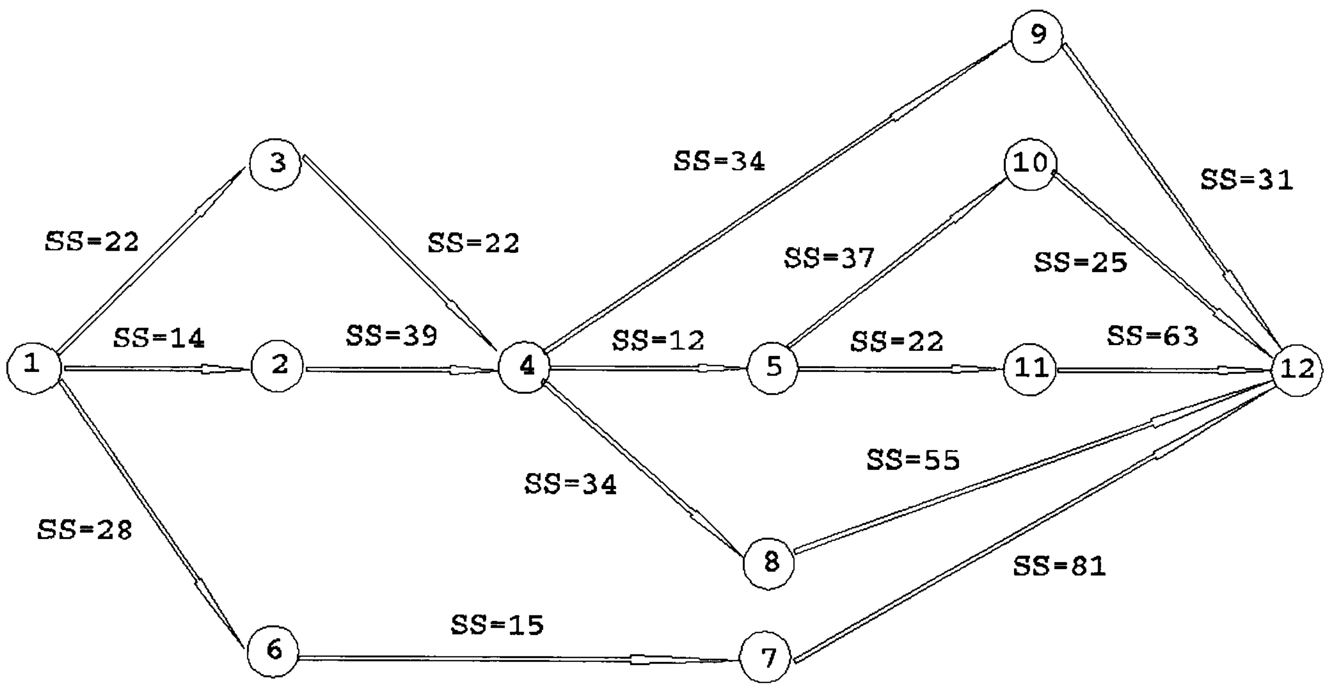
The MITC rule will be used as the backtracking rule for the objective of cost minimization because it yields a very good heuristic solution, and requires less computation time than the other rules. The MITR rule will be used as the backtracking rule for the resource leveling objective because it gives a good heuristic solution within a shorter time.

6.5 An Application of the Multiple Objective Algorithm to a Warehouse Project

The multiple objective algorithm is implemented to a modified version of an actual warehouse project taken from Barrie and Paulson (1992). The project was designed by Leo Rosenthal, A.I.A., a practicing Denver architect. The 151,600 sq.ft. warehouse project is intended to provide storage for grocery and nonfood items, equipped with pallet racks and flow racks. The modified network of the warehouse project is given in Figure 6.2. The fair-cost estimate of the project is summarized in Table 6.5, and scheduling information of the warehouse project is given in Table 6.6.

The warehouse project consists of twelve activities, including one dummy terminal activity. Each activity can be performed by using one of the three available modes of operation, except the dummy terminal activity having only one mode. Activity durations range from 22 to 138 time units. Activities compete for one renewable resource type (labor) and one nonrenewable resource type (money). Network complexity

FIGURE 6.2 A MODIFIED NETWORK OF A WAREHOUSE PROJECT



NOTES: (1) SS represents the start-to-start precedence relationship.

(2) Activity durations are in term of working days.

TABLE 6.5 FAIR-COST ESTIMATE SUMMARY FOR A WAREHOUSE PROJECT

Contract Package	Labor Hours	Labor Cost	Material Cost	Markup	Total Cost
1. Site Earthwork	1470	44000	153000	24000	221000
2. Concrete Work	7730	232000	349000	69000	650000
3. Special Floors	7330	220000	272000	59000	551000
3.1) Start	3420	103000	127000	27000	257000
3.2) Complete	3910	117000	145000	32000	294000
4. Structural Steel	3770	113000	686000	96000	895000
5. Precast Walls	2640	79000	418000	60000	557000
6. Plumbing & HVAC	4250	128000	313000	53000	494000
7. Fire Protection	3220	96000	201000	36000	333000
8. Electrical	3470	104000	250000	42000	396000
9. Roofing	3150	94000	124000	26000	244000
10. Building Finish	8470	255000	311000	68000	634000
Total	45500	1365000	3077000	533000	4975000

Average Labor Cost $ 30.00/hour (including taxes, insurance, and fringes)

Total Square Feet 151,600 sq.ft.

TABLE 6.6 SCHEDULING INFORMATION FOR A WAREHOUSE PROJECT

Job Description	Mode	Duration (work-days)	Labor (men)	Total Cost (US $)
1. Site Earthwork	1	46	4	215000
	2	62	3	221000
	3	92	2	232000
2. Concrete Work	1	88	11	638000
	2	108	9	650000
	3	138	7	670000
3. Start Special Floors	1	22	20	252000
	2	27	16	257000
	3	36	12	266000
4. Structural Steel	1	43	11	877000
	2	53	9	895000
	3	68	7	922000
5. Precast Walls	1	37	9	544000
	2	47	7	557000
	3	66	5	581000
6. Plumbing & HVAC	1	89	6	486000
	2	106	5	494000
	3	133	4	508000
7. Fire Protection	1	81	5	326000
	2	101	4	333000
	3	134	3	345000
8. Electrical	1	55	8	386000
	2	72	6	396000
	3	109	4	418000
9. Roofing	1	31	13	239000
	2	39	10	244000
	3	57	7	256000
10. Complete Special Floors	1	25	20	287000
	2	31	16	294000
	3	41	12	304000
11. Building Finish	1	63	17	619000
	2	82	13	634000
	3	118	9	664000
12. Facility Complete (Dummy Job)	1	0	0	0

(the ratio of the number of arcs to the number of jobs) for the problem is 1.33. Critical path early finish time assuming unlimited resources is 150 time units.

Several resource combinations are tested for the problem, including both loose (Test#1) and tight (Test#2) renewable resource limits. The test problems are described below.

Test#1: Job labeling rule = MAAD

Backtracking rule for time minimization = LIFO

Backtracking rule for cost minimization = MITC

Backtracking rule for resource leveling = MITR

Nonrenewable resource limit (Cost) = 5000000

Renewable resource limit (Labor) = 70

Maximum computation time = 30 minutes

Test#2: Job labeling rule = MAAD

Backtracking rule for time minimization = LIFO

Backtracking rule for cost minimization = MITC

Backtracking rule for resource leveling = MITR

Nonrenewable resource limit (Cost) = 5000000

Renewable resource limit (Labor) = 30

Maximum computation time = 30 minutes

The following combinations of priority levels are investigated for each of the test problems.

Case 1: Highest Priority Goal: Time Minimization

Second Highest Priority Goal: Cost Minimization

Lowest Priority Goal: Resource Leveling Objective

Case 2: Highest Priority Goal: Cost Minimization

Second Highest Priority Goal: Time Minimization

Lowest Priority Goal: Resource Leveling Objective

Case 3: Highest Priority Goal: Resource Leveling Objective

Second Highest Priority Goal: Time Minimization

Lowest Priority Goal: Cost Minimization

A summary of the test results is given in Table 6.7. The test results show that optimality is reached for both time and cost objectives within a second for the problem with loose resource limit (Test#1), and a good heuristic solution is obtained for the resource leveling objective within the specified computation time. For the problem with tight resource limit (Test#2), a good heuristic solution is obtained for each objective within the specified computation time, except that optimality is reached for the cost objective within a second when the cost objective is the highest priority goal.

The test results also show that there are no differences in the objective function values among different combinations of priority levels for the problem with loose resource limit (Test#1). For the problem with tight resource limit (Test#2), different objective function values are obtained when priority levels are changed.

Even though the problem has only 12 jobs and one type of each renewable and nonrenewable resource, the problem is still computational intensive when there is a tight renewable resource limit. This is because a large value of activity

TABLE 6.7 TEST RESULTS FOR A WAREHOUSE PROJECT

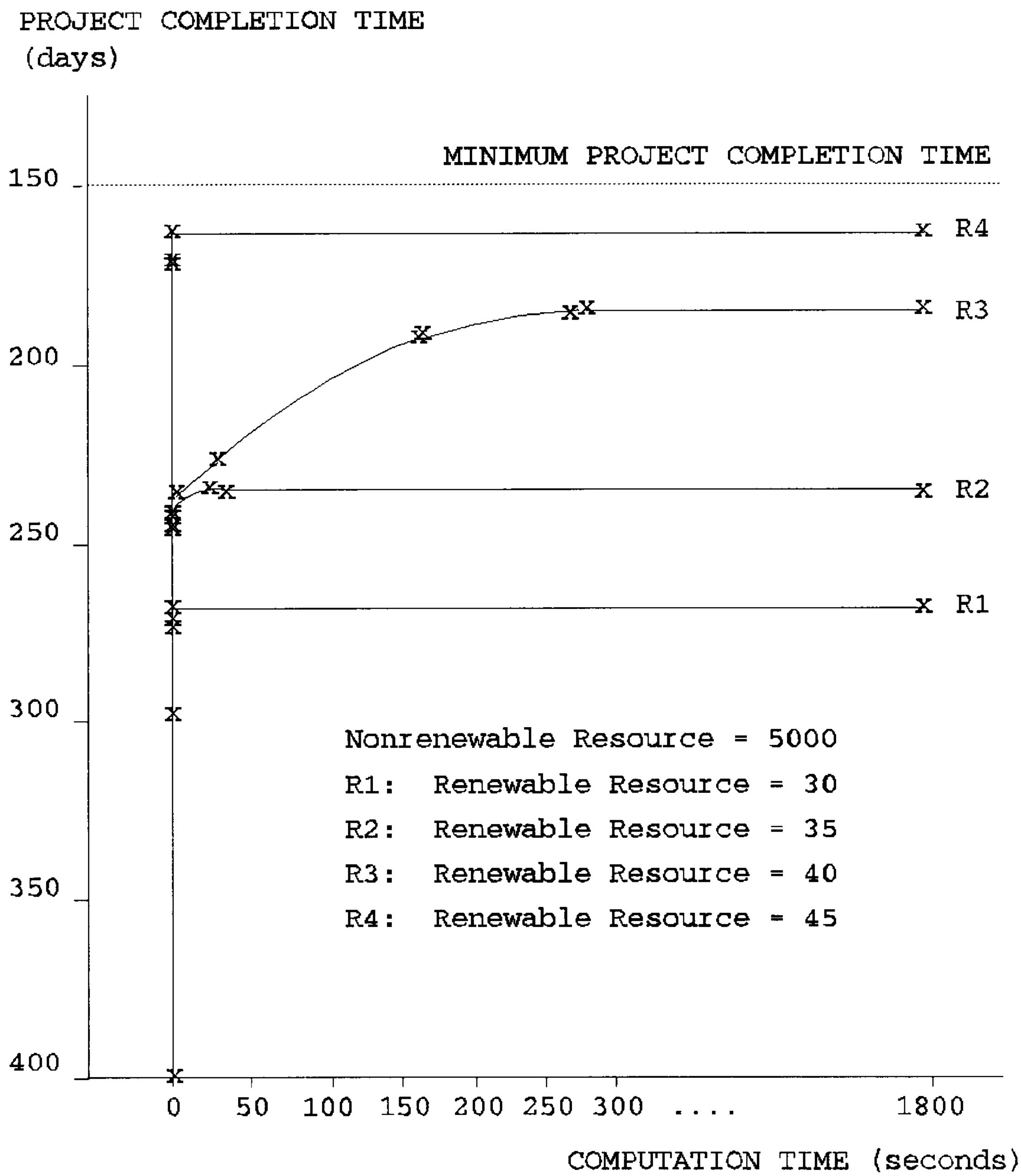
	Objective Values			Computation Time		
	Time (days)	Cost (1000$)	Level (units)	Time	Cost (seconds)	Level
Test#1						
Case 1:	150	4869	4749	< 1	< 1	*
Case 2:	150	4869	4749	< 1	< 1	*
Case 3:	150	4869	4749	< 1	< 1	*
Test#2						
Case 1:	259	4894	2023	*	*	*
Case 2:	274	4869	2469	*	< 1	*
Case 3:	247	4998	1708	*	*	*

Notes: 1. The (*) for the computation time in the above table means that an optimal solution cannot be found within the specified computation time (30 minutes).

2. The objective function values shown in the above table are the final values obtained after all goals are evaluated.

duration significantly increases the computation time required to obtain an optimal solution. However, a solution obtained from the problem which cannot reach optimality within the specified computation time is considered to be a good heuristic solution for the problem. Figure 6.3 shows the relationship between the objective function value and computation time as a function of renewable resources for the warehouse problem.

FIGURE 6.3 RELATIONSHIP BETWEEN OBJECTIVE FUNCTION VALUE AND COMPUTATION TIME AS A FUNCTION OF RENEWABLE RESOURCES



<u>NOTE</u>: The above relationship is based on the time minimization problem. Several renewable resource limitations are tested for the problem.

CHAPTER 7

CONCLUSIONS AND RECOMMENDATIONS

7.1 Usefulness and Contributions

The usefulness and contributions of this research are listed below:

(1) A specialized algorithm for the multi-objective resource-constrained project scheduling problem is developed. The algorithm gives an optimal solution to the problem using less computation time than the mathematical (integer) programming approaches.

(2) The algorithm is designed specifically to handle the preemptive goal programming resource-constrained project scheduling problem for time, cost, and resource leveling objectives. The algorithm allows users to determine the priority for each goal, and select whether the users want to obtain optimal solutions or heuristic solutions. It also gives an optimal value to the goal for which the desired level is unspecified.

(3) The algorithm is the first specialized optimization technique which includes the characteristics of splittable and nonsplittable jobs, renewable and nonrenewable resources, variation in resource availability, time-resource tradeoff (variation in resource consumption), time-cost tradeoff, and

multiple objectives.

(4) Algorithms for a single objective resource-constrained project scheduling problem are also provided for each of the three objective functions including time minimization, cost minimization, and resource leveling objectives. Like the multiple objective algorithm, these algorithms allow users to determine whether optimal solutions or heuristic solutions are desired.

(5) Improvements have been made over Talbot's algorithm which result in a significant reduction in computation time of the time minimization and cost minimization algorithms.

(6) The algorithm for the resource leveling problem is a specialized solution technique; use of similar algorithm has not been reported in published literature.

7.2 Limitations

The limitations of the algorithms are as follows:

(1) An activity-on-node diagram is used to describe the network, rather than an activity-on-arrow diagram.

(2) No scheduled or directed times are allowed on any intermediate job.

(3) A nonrenewable resource is assumed to be constrained only over the project life.

(4) The time-based bounding technique is used to determine the lower and upper bounds of activity completion times.

(5) Only the finish-to-start relationship without lead/lag time is allowed for splittable jobs.

(6) A splittable jobs must resume in the same mode of operation after the interruption.

(7) A schedule obtained from the algorithms is an early start (finish) schedule where activities are scheduled at their earliest start (completion) time.

(8) An upper bound approach is used in the single objective algorithm to reach an optimal solution.

(9) No special algorithm is provided to handle the multi-project problem.

(10) The algorithm can handle only the constraints resulting from controllable factors such as network logic, and time, cost and resource limitation.

7.3 Possible Areas for Further Research

The following recommendations are suggested to improve and enhance performance of the current algorithms.

(1) A special code should be provided for the algorithms to translate input information from activity-on-arrow network diagram to activity-on-node network diagram.

(2) The algorithms may be extended to include scheduled times on intermediate jobs, and allow a nonrenewable resource to be constrained not only over the project life, but also at some intervals such as on monthly or quarterly basis.

(3) To determine the lower and upper bounds of activity completion time, other technique such as the resource-based bounding technique (Patterson and Huber, 1974) may be tested.

(4) The algorithms may be extended to handle other types of precedence relationship with lead/lag time for spliitable jobs.

(5) A special algorithm should be developed to handle the case of splittable job where a job can be preempted a number of times, and can resume its operation by using a different mode after the interruption. A mathematical formulation was provided by Pritsker, Watters and Wolfe (1969); however, effective algorithms to solve such problem are not available.

(6) The algorithms can be modified to obtain a late start (finish) schedule where activities are scheduled at their latest start (completion) time. This will allow users to have a more flexible schedule. However, it is not necessary that a schedule between the early start (finish) schedule

and the late start (finish) schedule is feasible because the schedule has not been tested for resource feasibility.

(7) The proposed single objective algorithm can be easily modified, like Talbot's algorithm, to accommodate the horizon-varying process developed by Patterson and Huber (1974) which consists of the lower bound, upper bound, and binary search approaches.

(8) A special algorithm should be provided for the multi-project problem.

(9) The algorithms should be extended to handle both controllable factors and uncontrollable factors such as weather conditions, resource delays, and changed conditions. An algorithm proposed by Chang et al. (1990) using the concept of a fuzzy expert system could be modified and used as a guideline for more complex algorithms.

(10) Techniques similar to the network cut concept, introduced by Talbot (1978), could be used to increase computational efficiency of the algorithms.

REFERENCES

1. Ahuja, H., Construction Performance Control by Networks, John Wiley and Sons, New York, N.Y., 1976.

2. Antill, James M., and Woodhead, Ronald W., Critical Path Methods in Construction Practice, Second Edition, Wiley-Interscience, New York, N.Y., 1970.

3. Balas, Egon, "An Additive Algorithm for Solving Linear Programs with Zero-One Variables," Operations Research, 13, 4, 517-546, 1965.

4. Balas, Egon, "Bivalent Programming by Implicit Enumeration," Operations Research Support Methodology, Holzman, Albert G., (Ed.), Marcel Dekker, Inc., New York, N.Y., 1979.

5. Balinski, M. L., "Integer Programming: Methods, Uses, Computation," Management Science, 12, 3, 253-313, 1965.

6. Barrie, Donald S., and Paulson, Boyd C., Professional Construction Management: Including C.M., Design-Construct, and General Contracting, Third Edition, McGraw-Hill, Inc., New York, N.Y., 1992.

7. Bowman, E. H., "The Schedule-Sequencing Problem," Operations Research, 7, 5, 621-624, 1959.

8. Burgess, A. R., and Killebrew, J. B., "Variation in Activity Level on a Cyclic Arrow Diagram," Journal of Industrial Engineering, 13, 2, 76-83, 1962.

9. Brand, J. D., Meyer, W. L., and Shaffer, L. R., "The Resource Scheduling Problem in Construction," Civil Engineering Studies Report No. 5, Department of Civil Engineering, University of Illinois, Urbana, 1964.

10. Chang, T. C., Ibbs, C. W., and Crandall, K. C., "Network Resource Allocation with Support of a Fuzzy Expert System," Journal of Construction Engineering and Management, 116, 2, 239-260, 1990.

11. Davis, Edward W., "An Exact Algorithm for the Multiple Constrained-Resource Project Scheduling Problem," Ph.D. dissertation, Department of Administrative Sciences, Yale University, New Haven, 1968.

12. Davis, Edward W., and Heidorn, George E., "An Algorithm for Optimal Project Scheduling Under Multiple Resource Constraints," Management Science, 17, 12, B803-B816,

1971.

13. Davis, Edward W., "Project Scheduling under Resource Constraints-Historical Review and Categorization of Procedures," AIIE Transactions, 5, 4, 297-313, 1973.

14. Davis, Edward W., and Patterson, James H., "A Comparison of Heuristic and Optimum Solutions in Resource-Constrained Project Scheduling," Management Science, 21, 8, 944-955, 1975.

15. Easa, Said M., "Resource Leveling in Construction by Optimization," Journal of Construction Engineering and Management, 115, 2, 302-316, 1989.

16. Elmaghraby, Salah E., "The Sequencing of N Jobs on M Parallel Processors," unpublished paper, North Carolina State University, Raleigh, 1967.

17. Elmaghraby, Salah E., "The Sequencing of N Jobs on M Parallel Processors with Extensions to the Scarce Resources Problem of Activity Networks," Proceedings of the Inaugural Conference of the Scientific Computation Center and Institute of Statistical Studies and Research, Cairo, Egypt, 230-255, 1969.

18. Elmaghraby, Salah, E., Activity Networks: Project Planning and Control by Network Models, John Wiley and Sons, New York, N.Y., 1977.

19. Galbreath, R., "Computer Program for Leveling Resource Usage," Journal of Construction Div., ASCE, 91, 1, 107-124, 1965.

20. Geoffrion, A. M., "An Improved Implicit Enumeration Approach for Integer Programming," Operations Research, 17, 3, 437-454, 1969.

21. Gutjahr, Allan L., "An Algorithm for the Assembly-Line Balancing Problem," unpublished MS thesis, Johns Hopkins University, Baltimore, 1963.

22. Gutjahr, Allan L., and Nemhauser, George L., "An Algorithm for the Line Balancing Problem," Management Science, 11, 2, 308-315, 1964.

23. Hadley, George, "Project Planning and Manpower Scheduling," Sec. 8.8 of Nonlinear and Dynamic Programming, Addison-Wesley Publishing Company, Inc., 1964.

24. Hall, John R. Jr., (Ed.), "Management Science Update

Column: Multiproject Scheduling through Zero-One Programming," Management Science, 26, 2, 228-229, 1980.

25. Hannan, Edward L., "The Application of Goal Programming techniques to the CPM Problem," Socio-Economic Planning Sciences, 12, 5, 267-270, 1978.

26. Harris, Robert B., Precedence and Arrow Networking Techniques for Construction, John Wiley and Sons, New York, N.Y., 1978.

27. Johnson, Thomas J. R., "An Algorithm for the Resource-Constrained Project Scheduling Problem," Unpublished Ph.D. dissertation, School of Management, Massachusetts Institute of Technology, 1967.

28. Karp, R., "On Computational Complexity of Combinatorial Problems," Networks, 5, 1, 45-68, 1975.

29. Kurtulus, Ibrahim S., and Narula, Subhash C., "Multi-Project Scheduling: Analysis of Project Performance," IIE Transactions, 17, 1, 58-66, 1985.

30. Lee, Sang M., Moore, Laurence J., and Clayton, Edward R., "Analysis of Multi-Objective Project Crashing Models," 8th Annual Conference, American Institute for Decision Sciences Proceedings (San Francisco: AIDS), November 10-12, 1976, pp. 114-115.

31. Lee, Sang M., Goal Programming Methods for Multiple Objective Integer Programs. Monograph, Operations Research Division, American Institute of Industrial Engineers, 1979.

32. Lee, Sang M., and Olson, David L., "A Zero-One Goal Programming Approach to Multi-Project Scheduling," Project Management Journal, 15, 2, 61-71, 1984.

33. Lee, Sang M., and Olson, David L., "Project Scheduling for Multiple Objectives," In Project Management: Methods and Studies, Dean, B. V. (ed.), Elsevier Science: North-Holland Publishing Company, 1985.

34. Lenstra, J., and Rinnooy Kan, A., "New Directions in Scheduling Theory," Operation Research Letters, 2, 6, 255-259, 1984.

35. Mandeville, D. W., "The Development of Network Analysis Resource Balancing Methods from Assembly Line Balancing Techniques," unpublished MS thesis, Purdue University, Lafayette, Indiana, 1965.

36. Moder, Joseph J., Phillips, Cecil R., and Davis, Edward W., Project Management with CPM, PERT and Precedence Diagramming, Third Edition, Van Nostrand Reinhold Company, New York, N.Y., 1983.

37. Mohanty, R. P., and Siddiq, M. K., "Multiple projects-multiple resources-constrained scheduling: some studies," International Journal of Production Research, 27, 2, 261-280, 1989.

38. Moodie, C. L., and Mandeville, D. W., "Project Resource Balancing by Assembly Line Balancing Techniques," Journal of Industrial Engineering, July, 1965.

39. Norbis, Mario I., and Smith, J. MacGregor, "A Multiobjective, Multi-Level Heuristic for Dynamic Resource Constrained Scheduling Problems," European Journal of Operational Research, 33, 1, 30-41, 1988.

40. Ostle, Bernard, and Malone, Linda C., Statistics in Research: Basic Concepts and Techniques for Research Workers, Forth Edition, Iowa State University Press, Ames, Iowa, 1988.

41. Patterson, James H., "Alternate Methods of Project Scheduling with Limited Resources," Naval Research Logistics Quarterly, 20, 4, 767-784, 1973.

42. Patterson, James H., and Huber, Walter D., "A Horizon-Varying, Zero-One Approach to Project Scheduling," Management Science, 20, 6, 990-998, 1974.

43. Patterson, James H., and Roth, Glenn W., "Scheduling a Project Under Multiple Resource Constraints: A Zero-One Programming Approach," AIIE Transactions, 8, 4, 449-455, 1976.

44. Patterson, James H., "A Comparison of Exact Approaches for Solving the Multiple Constrained Resource, Project Scheduling Problem," Management Science, 30, 7, 854-867, 1984.

45. Pritsker, A. Alan B., Watters, Lawrence J., and Wolfe, Philip M., "Multiproject Scheduling with Limited Resources: A Zero-One Programming Approach," Management Science, 16, 1, 93-108, 1969.

46. Shaffer, L., Ritter, J., and Mayer, W., The Critical Path Method, McGraw-Hill Book Co., New York, N.Y., 1965.

47. Slowinski, Roman, "Multiobjective Network Scheduling with Efficient Use of Renewable and Nonrenewable Resources,"

European Journal of Operational Research, 7, 3, 265-273, 1981.

48. Stinson, Joel P., "A Branch and Bound Algorithm for a General Class of Resource-Constrained Scheduling Problems," Ph.D. dissertation, The University of North Carolina at Chapel Hill, 1976.

49. Stinson, Joel P., Davis, Edward W., and Khumawala, Basheer M., "Multiple Resource-Constrained Scheduling Using Branch and Bound," AIIE Transactions, 10, 3, 252-259, 1978.

50. Talbot, F. Brian, "An Integer Programming Algorithm for the Resource-Constrained Project Scheduling Problem," Ph.D. dissertation, Pennsylvania State University, 1976.

51. Talbot, F. Brian, and Patterson, James H., "An Efficient Integer Programming Algorithm with Network Cuts for Solving Resource-Constrained Scheduling Problems," Management Science, 24, 11, 1163-1174, 1978.

52. Talbot, F. Brian, "Resource-Constrained Project Scheduling with Time-Resource Tradeoffs: The Nonpreemptive Case," Management Science, 28, 10, 1197-1210, 1982.

53. Wiest, Jerome D., "The Scheduling of Large Projects with Limited Resources," unpublished Ph.D. thesis, Carnegie Institute of Technology, 1963.

54. Wiest, Jerome D., and Levy, Ferdinand K., A Management Guide to PERT/CPM, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1977.

55. Wieters, C. David, "Allocation of Resources Among Competing Projects Using CPM and Integer Goal Programming," 11th Annual Meeting, American Institute for Decision Sciences, November 19-21, 1979, New Orleans Hilton, New Orleans, pp. 371-373.

56. Woodworth, Bruce M., and Willie, Charles J., "A Heuristic Algorithm for Resource Leveling in Multi-Resource Scheduling," Decision Sciences, 6, 3, 525-540, 1975.

APPENDICES

APPENDIX 1

PROBABILITY DISTRIBUTIONS

The probability distribution of job durations used by Davis (1968) and Johnson (1967) is given below.

Duration	1	2	3	4	5	6	7	8	9
Probability	.156	.231	.186	.122	.097	.079	.060	.041	.028

Based on the above probability distribution, the probability distributions of modes of operation and of job durations can be obtained as follows:

1. Probability Distribution of Modes of Operation.

Let mode 1 represent the job durations of 1, 2 and 3 time units, mode 2 represent the job durations of 4, 5 and 6 time units, and mode 3 represent the job durations of 7, 8 and 9 time units. Thus, if a job has only one mode of operation; the probability that mode 1 will be selected is equal to the sum of probability of job durations of 1, 2 and 3 time units (.573), the probability that mode 2 will be selected is equal to the sum of probability of job durations of 4, 5 and 6 time units (.298), and the probability that mode 3 will be selected is equal to the sum of probability of job durations of 7, 8 and 9 time units (.129). If a job has two modes of operation, the probability of selecting any combination of two modes is equal to the sum of probability of each mode divided by 2. This is because the combinations are independent of each other. If a job has three modes of operation, all modes must be selected because the maximum number of modes allowed in this study is three.

2. Probability Distribution of Job Durations.

Since mode 1 consists of job durations of 1, 2 and 3 time units; thus, the probability that job duration is equal to 1 within mode 1 is equal to the probability of job duration of 1 divided by the probability of mode 1. Probability of the others can be obtained by using the same technique which is shown below.

MODE	PROBABILITY	DURATION	PROB.(DURATION/MODE)
1	.573	1	.156/.573 = .272
		2	.231/.573 = .403
		3	.186/.573 = .325
2	.298	4	.122/.298 = .409
		5	.097/.298 = .326
		6	.079/.298 = .265
3	.129	7	.060/.129 = .465
		8	.041/.129 = .318
		9	.028/.129 = .217

The probability distribution of renewable resources consumption used in this study is the same as the one used by Davis (1968) and Johnson (1967).

APPENDIX 2

TEST PROBLEMS

PROBLEM 1: 10 JOBS, SHAPE#1(1), LOW COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	3	1	0	0	2	3	3
1	2	0	0	0	0	0	0	0
1	3	0	0	0	0	0	0	0
2	1	1	1	0	0	2	1	1
2	2	6	0	1	0	2	1	1
2	3	8	0	0	1	2	1	1
3	1	1	1	0	0	1	5	2
3	2	4	0	1	0	1	5	2
3	3	7	0	0	1	1	5	2
4	1	2	1	0	0	2	5	3
4	2	4	0	1	0	2	5	3
4	3	7	0	0	1	2	5	3
5	1	0	0	0	0	0	0	0
5	2	4	0	1	0	1	3	3
5	3	0	0	0	0	0	0	0
6	1	1	1	0	0	3	1	6
6	2	0	0	0	0	0	0	0
6	3	0	0	0	0	0	0	0
7	1	0	0	0	0	0	0	0
7	2	5	0	1	0	4	3	4
7	3	0	0	0	0	0	0	0
8	1	2	1	0	0	2	2	2
8	2	0	0	0	0	0	0	0
8	3	0	0	0	0	0	0	0
9	1	0	0	0	0	0	0	0
9	2	0	0	0	0	0	0	0
9	3	8	0	0	1	5	6	1
10	1	2	1	0	0	6	1	3
10	2	4	0	1	0	6	1	3
10	3	7	0	0	1	6	1	3

PRECEDENCE RELATIONSHIPS

JOB 1 precedes: JOB 2 JOB 3
JOB 2 precedes: JOB 5 JOB 6
JOB 3 precedes: JOB 4 JOB 5
JOB 4 precedes: JOB 7 JOB 9
JOB 5 precedes: JOB 8
JOB 6 precedes: JOB 7

JOB 7 precedes: JOB 10
JOB 8 precedes: JOB 10
JOB 9 precedes: JOB 10
JOB 10 precedes: -

NOTE: [1] refer to the shape number given in Table 6.2

PROBLEM 2: 10 JOBS, SHAPE#1, MEDIUM COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	2	1	0	0	1	3	3
1	2	5	0	1	0	1	3	3
1	3	9	0	0	1	1	3	3
2	1	3	1	0	0	2	4	5
2	2	6	0	1	0	2	4	5
2	3	0	0	0	0	0	0	0
3	1	0	0	0	0	0	0	0
3	2	5	0	1	0	2	5	2
3	3	0	0	0	0	0	0	0
4	1	3	1	0	0	3	1	1
4	2	4	0	1	0	3	1	1
4	3	9	0	0	1	3	1	1
5	1	2	1	0	0	3	5	2
5	2	0	0	0	0	0	0	0
5	3	0	0	0	0	0	0	0
6	1	3	1	0	0	5	5	3
6	2	4	0	1	0	5	5	3
6	3	8	0	0	1	5	5	3
7	1	0	0	0	0	0	0	0
7	2	5	0	1	0	5	1	5
7	3	8	0	0	1	5	1	5
8	1	2	1	0	0	1	1	2
8	2	6	0	1	0	1	1	2
8	3	0	0	0	0	0	0	0
9	1	2	1	0	0	3	6	3
9	2	0	0	0	0	0	0	0
9	3	0	0	0	0	0	0	0
10	1	1	1	0	0	3	2	2
10	2	4	0	1	0	3	2	2
10	3	0	0	0	0	0	0	0

PRECEDENCE RELATIONSHIPS

JOB 1 precedes: JOB 2 JOB 3
JOB 2 precedes: JOB 5 JOB 6
JOB 3 precedes: JOB 4
JOB 4 precedes: JOB 7 JOB 8
JOB 5 precedes: JOB 7 JOB 8 JOB 9
JOB 6 precedes: JOB 7 JOB 8 JOB 9
JOB 7 precedes: JOB 10
JOB 8 precedes: JOB 10
JOB 9 precedes: JOB 10
JOB 10 precedes: -

PROBLEM 3: 10 JOBS, SHAPE#1, HIGH COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION					
			TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	0	0	0	0	0	0	0
1	2	0	0	0	0	0	0	0
1	3	8	0	0	1	1	4	3
2	1	2	1	0	0	2	4	4
2	2	0	0	0	0	0	0	0
2	3	0	0	0	0	0	0	0
3	1	2	1	0	0	2	2	4
3	2	5	0	1	0	2	2	4
3	3	0	0	0	0	0	0	0
4	1	2	1	0	0	2	3	1
4	2	0	0	0	0	0	0	0
4	3	8	0	0	1	2	3	1
5	1	3	1	0	0	5	1	3
5	2	4	0	1	0	5	1	3
5	3	7	0	0	1	5	1	3
6	1	0	0	0	0	0	0	0
6	2	0	0	0	0	0	0	0
6	3	7	0	0	1	2	2	3
7	1	3	1	0	0	3	3	3
7	2	5	0	1	0	3	3	3
7	3	7	0	0	1	3	3	3
8	1	2	1	0	0	6	4	2
8	2	4	0	1	0	6	4	2
8	3	9	0	0	1	6	4	2
9	1	3	1	0	0	1	3	2
9	2	6	0	1	0	1	3	2
9	3	9	0	0	1	1	3	2
10	1	3	1	0	0	4	4	2
10	2	0	0	0	0	0	0	0
10	3	0	0	0	0	0	0	0

PRECEDENCE RELATIONSHIPS

JOB 1 precedes:	JOB 2	JOB 3	
JOB 2 precedes:	JOB 4	JOB 5	JOB 6
JOB 3 precedes:	JOB 4	JOB 5	JOB 6
JOB 4 precedes:	JOB 7	JOB 8	JOB 9
JOB 5 precedes:	JOB 7	JOB 9	
JOB 6 precedes:	JOB 7	JOB 9	
JOB 7 precedes:	JOB 10		
JOB 8 precedes:	JOB 10		
JOB 9 precedes:	JOB 10		
JOB 10 precedes:	-		

PROBLEM 4: 10 JOBS, SHAPE#2, LOW COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	3	1	0	0	2	1	4
1	2	5	0	1	0	2	1	4
1	3	0	0	0	0	0	0	0
2	1	1	1	0	0	1	6	3
2	2	4	0	1	0	1	6	3
2	3	0	0	0	0	0	0	0
3	1	3	1	0	0	5	2	1
3	2	0	0	0	0	0	0	0
3	3	0	0	0	0	0	0	0
4	1	2	1	0	0	3	4	1
4	2	4	0	1	0	3	4	1
4	3	9	0	0	1	3	4	1
5	1	3	1	0	0	2	3	3
5	2	0	0	0	0	0	0	0
5	3	0	0	0	0	0	0	0
6	1	3	1	0	0	1	2	4
6	2	4	0	1	0	1	2	4
6	3	7	0	0	1	1	2	4
7	1	3	1	0	0	6	2	2
7	2	4	0	1	0	6	2	2
7	3	7	0	0	1	6	2	2
8	1	2	1	0	0	5	5	4
8	2	0	0	0	0	0	0	0
8	3	9	0	0	1	5	5	4
9	1	1	1	0	0	5	2	3
9	2	4	0	1	0	5	2	3
9	3	7	0	0	1	5	2	3
10	1	2	1	0	0	5	2	2
10	2	5	0	1	0	5	2	2
10	3	7	0	0	1	5	2	2

PRECEDENCE RELATIONSHIPS

JOB 1 precedes: JOB 2 JOB 3 JOB 4 JOB 5
JOB 2 precedes: JOB 8
JOB 3 precedes: JOB 6
JOB 4 precedes: JOB 6 JOB 7
JOB 5 precedes: JOB 9
JOB 6 precedes: JOB 10
JOB 7 precedes: JOB 10
JOB 8 precedes: JOB 10
JOB 9 precedes: JOB 10
JOB 10 precedes: -

PROBLEM 5: 10 JOBS, SHAPE#2, MEDIUM COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	0	0	0	0	0	0	0
1	2	5	0	1	0	2	3	3
1	3	0	0	0	0	0	0	0
2	1	2	1	0	0	2	2	4
2	2	4	0	1	0	2	2	4
2	3	9	0	0	1	2	2	4
3	1	2	1	0	0	1	6	4
3	2	0	0	0	0	0	0	0
3	3	0	0	0	0	0	0	0
4	1	3	1	0	0	4	3	1
4	2	4	0	1	0	4	3	1
4	3	8	0	0	1	4	3	1
5	1	2	1	0	0	5	3	5
5	2	0	0	0	0	0	0	0
5	3	8	0	0	1	5	3	5
6	1	3	1	0	0	1	2	1
6	2	0	0	0	0	0	0	0
6	3	8	0	0	1	1	2	1
7	1	3	1	0	0	3	2	1
7	2	0	0	0	0	0	0	0
7	3	7	0	0	1	3	2	1
8	1	3	1	0	0	2	2	3
8	2	0	0	0	0	0	0	0
8	3	7	0	0	1	2	2	3
9	1	2	1	0	0	1	3	5
9	2	0	0	0	0	0	0	0
9	3	8	0	0	1	1	3	5
10	1	2	1	0	0	3	6	5
10	2	5	0	1	0	3	6	5
10	3	0	0	0	0	0	0	0

PRECEDENCE RELATIONSHIPS

JOB 1 precedes:	JOB 2	JOB 3	JOB 4	JOB 5
JOB 2 precedes:	JOB 6	JOB 7	JOB 8	JOB 9
JOB 3 precedes:	JOB 6	JOB 7	JOB 9	
JOB 4 precedes:	JOB 8			
JOB 5 precedes:	JOB 8			
JOB 6 precedes:	JOB 10			
JOB 7 precedes:	JOB 10			
JOB 8 precedes:	JOB 10			
JOB 9 precedes:	JOB 10			
JOB 10 precedes:	-			

PROBLEM 6: 10 JOBS, SHAPE#2, HIGH COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	3	1	0	0	3	1	2
1	2	6	0	1	0	3	1	2
1	3	8	0	0	1	3	1	2
2	1	3	1	0	0	2	3	5
2	2	6	0	1	0	2	3	5
2	3	0	0	0	0	0	0	0
3	1	1	1	0	0	2	1	1
3	2	0	0	0	0	0	0	0
3	3	7	0	0	1	2	1	1
4	1	0	0	0	0	0	0	0
4	2	5	0	1	0	1	6	4
4	3	0	0	0	0	0	0	0
5	1	2	1	0	0	3	4	4
5	2	0	0	0	0	0	0	0
5	3	9	0	0	1	3	4	4
6	1	0	0	0	0	0	0	0
6	2	5	0	1	0	6	1	2
6	3	8	0	0	1	6	1	2
7	1	0	0	0	0	0	0	0
7	2	6	0	1	0	3	2	1
7	3	0	0	0	0	0	0	0
8	1	3	1	0	0	2	1	4
8	2	5	0	1	0	2	1	4
8	3	0	0	0	0	0	0	0
9	1	3	1	0	0	3	3	4
9	2	4	0	1	0	3	3	4
9	3	9	0	0	1	3	3	4
10	1	2	1	0	0	3	3	2
10	2	0	0	0	0	0	0	0
10	3	0	0	0	0	0	0	0

PRECEDENCE RELATIONSHIPS

JOB 1 precedes:	JOB 2	JOB 3	JOB 4	JOB 5
JOB 2 precedes:	JOB 7	JOB 9		
JOB 3 precedes:	JOB 7	JOB 8	JOB 9	
JOB 4 precedes:	JOB 6	JOB 7	JOB 8	JOB 9
JOB 5 precedes:	JOB 6	JOB 7	JOB 8	JOB 9
JOB 6 precedes:	JOB 10			
JOB 7 precedes:	JOB 10			
JOB 8 precedes:	JOB 10			
JOB 9 precedes:	JOB 10			
JOB 10 precedes:	-			

PROBLEM 7: 20 JOBS, SHAPE#1, LOW COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	0	0	0	0	0	0	0
1	2	0	0	0	0	0	0	0
1	3	7	0	0	1	1	4	3
2	1	3	1	0	0	4	3	1
2	2	0	0	0	0	0	0	0
2	3	7	0	0	1	4	3	1
3	1	0	0	0	0	0	0	0
3	2	0	0	0	0	0	0	0
3	3	7	0	0	1	2	3	2
4	1	1	1	0	0	3	1	1
4	2	5	0	1	0	3	1	1
4	3	0	0	0	0	0	0	0
5	1	0	0	0	0	0	0	0
5	2	6	0	1	0	3	2	3
5	3	0	0	0	0	0	0	0
6	1	2	1	0	0	5	1	1
6	2	5	0	1	0	5	1	1
6	3	7	0	0	1	5	1	1
7	1	3	1	0	0	1	2	4
7	2	0	0	0	0	0	0	0
7	3	8	0	0	1	1	2	4
8	1	3	1	0	0	6	1	2
8	2	5	0	1	0	6	1	2
8	3	7	0	0	1	6	1	2
9	1	2	1	0	0	1	3	1
9	2	4	0	1	0	1	3	1
9	3	7	0	0	1	1	3	1
10	1	1	1	0	0	3	4	3
10	2	5	0	1	0	3	4	3
10	3	0	0	0	0	0	0	0
11	1	3	1	0	0	4	6	5
11	2	4	0	1	0	4	6	5
11	3	8	0	0	1	4	6	5
12	1	1	1	0	0	5	3	5
12	2	0	0	0	0	0	0	0
12	3	0	0	0	0	0	0	0
13	1	2	1	0	0	2	2	4
13	2	5	0	1	0	2	2	4
13	3	0	0	0	0	0	0	0
14	1	2	1	0	0	4	1	4
14	2	5	0	1	0	4	1	4
14	3	0	0	0	0	0	0	0
15	1	0	0	0	0	0	0	0
15	2	4	0	1	0	2	5	3
15	3	0	0	0	0	0	0	0

PROBLEM 7 (continued)

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION					
			TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
16	1	0	0	0	0	0	0	0
16	2	6	0	1	0	4	1	3
16	3	0	0	0	0	0	0	0
17	1	2	1	0	0	4	4	2
17	2	5	0	1	0	4	4	2
17	3	7	0	0	1	4	4	2
18	1	3	1	0	0	2	2	5
18	2	0	0	0	0	0	0	0
18	3	0	0	0	0	0	0	0
19	1	3	1	0	0	5	4	2
19	2	0	0	0	0	0	0	0
19	3	9	0	0	1	5	4	2
20	1	0	0	0	0	0	0	0
20	2	0	0	0	0	0	0	0
20	3	8	0	0	1	2	5	4

PRECEDENCE RELATIONSHIPS

JOB 1 precedes: JOB 2 JOB 3 JOB 4
JOB 2 precedes: JOB 7 JOB 8
JOB 3 precedes: JOB 5 JOB 6 JOB 7
JOB 4 precedes: JOB 7 JOB 9
JOB 5 precedes: JOB 10 JOB 11
JOB 6 precedes: JOB 10
JOB 7 precedes: JOB 10 JOB 11 JOB 13 JOB 14
JOB 8 precedes: JOB 11
JOB 9 precedes: JOB 12
JOB 10 precedes: JOB 18
JOB 11 precedes: JOB 15 JOB 18
JOB 12 precedes: JOB 17
JOB 13 precedes: JOB 16 JOB 19
JOB 14 precedes: JOB 18 JOB 19
JOB 15 precedes: JOB 20
JOB 16 precedes: JOB 20
JOB 17 precedes: JOB 20
JOB 18 precedes: JOB 20
JOB 19 precedes: JOB 20
JOB 20 precedes: -

PROBLEM 8: 20 JOBS, SHAPE#1, MEDIUM COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	3	1	0	0	5	1	2
1	2	5	0	1	0	5	1	2
1	3	0	0	0	0	0	0	0
2	1	2	1	0	0	3	2	3
2	2	0	0	0	0	0	0	0
2	3	0	0	0	0	0	0	0
3	1	2	1	0	0	3	2	5
3	2	4	0	1	0	3	2	5
3	3	8	0	0	1	3	2	5
4	1	0	0	0	0	0	0	0
4	2	6	0	1	0	2	4	4
4	3	0	0	0	0	0	0	0
5	1	2	1	0	0	1	2	3
5	2	0	0	0	0	0	0	0
5	3	0	0	0	0	0	0	0
6	1	0	0	0	0	0	0	0
6	2	6	0	1	0	4	3	3
6	3	7	0	0	1	4	3	3
7	1	3	1	0	0	3	1	1
7	2	4	0	1	0	3	1	1
7	3	9	0	0	1	3	1	1
8	1	2	1	0	0	3	3	1
8	2	5	0	1	0	3	3	1
8	3	7	0	0	1	3	3	1
9	1	2	1	0	0	3	1	1
9	2	0	0	0	0	0	0	0
9	3	0	0	0	0	0	0	0
10	1	3	1	0	0	1	3	3
10	2	4	0	1	0	1	3	3
10	3	0	0	0	0	0	0	0
11	1	1	1	0	0	2	1	5
11	2	5	0	1	0	2	1	5
11	3	9	0	0	1	2	1	5
12	1	1	1	0	0	1	5	1
12	2	6	0	1	0	1	5	1
12	3	7	0	0	1	1	5	1
13	1	0	0	0	0	0	0	0
13	2	0	0	0	0	0	0	0
13	3	7	0	0	1	5	2	3
14	1	2	1	0	0	4	2	5
14	2	5	0	1	0	4	2	5
14	3	9	0	0	1	4	2	5
15	1	3	1	0	0	2	1	3
15	2	0	0	0	0	0	0	0
15	3	9	0	0	1	2	1	3

PROBLEM 8 (continued)

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
16	1	3	1	0	0	3	2	2
16	2	4	0	1	0	3	2	2
16	3	9	0	0	1	3	2	2
17	1	2	1	0	0	4	1	2
17	2	0	0	0	0	0	0	0
17	3	0	0	0	0	0	0	0
18	1	0	0	0	0	0	0	0
18	2	4	0	1	0	2	2	1
18	3	0	0	0	0	0	0	0
19	1	1	1	0	0	2	5	1
19	2	5	0	1	0	2	5	1
19	3	0	0	0	0	0	0	0
20	1	0	0	0	0	0	0	0
20	2	5	0	1	0	5	3	4
20	3	0	0	0	0	0	0	0

PRECEDENCE RELATIONSHIPS

JOB 1 precedes:	JOB 2	JOB 3	JOB 4		
JOB 2 precedes:	JOB 5	JOB 6	JOB 7	JOB 8	JOB 9
JOB 3 precedes:	JOB 9				
JOB 4 precedes:	JOB 7				
JOB 5 precedes:	JOB 10	JOB 12	JOB 13	JOB 14	
JOB 6 precedes:	JOB 10	JOB 11	JOB 13	JOB 14	
JOB 7 precedes:	JOB 11	JOB 13	JOB 14		
JOB 8 precedes:	JOB 12	JOB 13	JOB 14		
JOB 9 precedes:	JOB 13				
JOB 10 precedes:	JOB 19				
JOB 11 precedes:	JOB 15				
JOB 12 precedes:	JOB 15	JOB 16	JOB 17		
JOB 13 precedes:	JOB 15	JOB 17			
JOB 14 precedes:	JOB 15	JOB 17	JOB 18	JOB 19	
JOB 15 precedes:	JOB 20				
JOB 16 precedes:	JOB 20				
JOB 17 precedes:	JOB 20				
JOB 18 precedes:	JOB 20				
JOB 19 precedes:	JOB 20				
JOB 20 precedes:	-				

PROBLEM 9: 20 JOBS, SHAPE#2, HIGH COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	0	0	0	0	0	0	0
1	2	5	0	1	0	1	1	1
1	3	0	0	0	0	0	0	0
2	1	2	1	0	0	1	1	3
2	2	0	0	0	0	0	0	0
2	3	0	0	0	0	0	0	0
3	1	1	1	0	0	3	4	1
3	2	0	0	0	0	0	0	0
3	3	0	0	0	0	0	0	0
4	1	3	1	0	0	3	5	3
4	2	4	0	1	0	3	5	3
4	3	0	0	0	0	0	0	0
5	1	0	0	0	0	0	0	0
5	2	4	0	1	0	1	1	2
5	3	8	0	0	1	1	1	2
6	1	2	1	0	0	4	1	3
6	2	0	0	0	0	0	0	0
6	3	7	0	0	1	4	1	3
7	1	3	1	0	0	5	2	4
7	2	0	0	0	0	0	0	0
7	3	0	0	0	0	0	0	0
8	1	0	0	0	0	0	0	0
8	2	4	0	1	0	3	3	6
8	3	0	0	0	0	0	0	0
9	1	0	0	0	0	0	0	0
9	2	5	0	1	0	2	6	5
9	3	8	0	0	1	2	6	5
10	1	1	1	0	0	2	4	3
10	2	5	0	1	0	2	4	3
10	3	9	0	0	1	2	4	3
11	1	0	0	0	0	0	0	0
11	2	6	0	1	0	3	2	1
11	3	8	0	0	1	3	2	1
12	1	2	1	0	0	2	1	1
12	2	0	0	0	0	0	0	0
12	3	9	0	0	1	2	1	1
13	1	2	1	0	0	2	3	1
13	2	6	0	1	0	2	3	1
13	3	9	0	0	1	2	3	1
14	1	2	1	0	0	2	2	6
14	2	0	0	0	0	0	0	0
14	3	0	0	0	0	0	0	0
15	1	3	1	0	0	2	4	4
15	2	6	0	1	0	2	4	4
15	3	7	0	0	1	2	4	4

PROBLEM 9 (continued)

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
16	1	1	1	0	0	5	2	1
16	2	0	0	0	0	0	0	0
16	3	0	0	0	0	0	0	0
17	1	1	1	0	0	5	4	3
17	2	0	0	0	0	0	0	0
17	3	8	0	0	1	5	4	3
18	1	1	1	0	0	3	2	4
18	2	4	0	1	0	3	2	4
18	3	0	0	0	0	0	0	0
19	1	1	1	0	0	4	3	2
19	2	0	0	0	0	0	0	0
19	3	8	0	0	1	4	3	2
20	1	0	0	0	0	0	0	0
20	2	5	0	1	0	4	4	2
20	3	0	0	0	0	0	0	0

PRECEDENCE RELATIONSHIPS

JOB 1 precedes:	JOB 2	JOB 3	JOB 4		
JOB 2 precedes:	JOB 6	JOB 8	JOB 9		
JOB 3 precedes:	JOB 5	JOB 6	JOB 7	JOB 8	
JOB 4 precedes:	JOB 5	JOB 6	JOB 7	JOB 9	
JOB 5 precedes:	JOB 11	JOB 12	JOB 14		
JOB 6 precedes:	JOB 10	JOB 11	JOB 12	JOB 13	JOB 14
JOB 7 precedes:	JOB 10	JOB 11	JOB 12	JOB 13	
JOB 8 precedes:	JOB 10	JOB 11	JOB 12	JOB 13	JOB 14
JOB 9 precedes:	JOB 10	JOB 12	JOB 13	JOB 14	
JOB 10 precedes:	JOB 17	JOB 18	JOB 19		
JOB 11 precedes:	JOB 15	JOB 16	JOB 17	JOB 18	JOB 19
JOB 12 precedes:	JOB 15	JOB 16	JOB 17	JOB 18	JOB 19
JOB 13 precedes:	JOB 15	JOB 17	JOB 18	JOB 19	
JOB 14 precedes:	JOB 18	JOB 19			
JOB 15 precedes:	JOB 20				
JOB 16 precedes:	JOB 20				
JOB 17 precedes:	JOB 20				
JOB 18 precedes:	JOB 20				
JOB 19 precedes:	JOB 20				
JOB 20 precedes:	-				

PROBLEM 10: 20 JOBS, SHAPE#2, LOW COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	1	1	0	0	2	3	3
1	2	4	0	1	0	2	3	3
1	3	7	0	0	1	2	3	3
2	1	2	1	0	0	6	4	2
2	2	4	0	1	0	6	4	2
2	3	8	0	0	1	6	4	2
3	1	1	1	0	0	5	5	2
3	2	0	0	0	0	0	0	0
3	3	0	0	0	0	0	0	0
4	1	3	1	0	0	1	2	5
4	2	5	0	1	0	1	2	5
4	3	9	0	0	1	1	2	5
5	1	3	1	0	0	1	4	5
5	2	0	0	0	0	0	0	0
5	3	0	0	0	0	0	0	0
6	1	2	1	0	0	4	2	3
6	2	0	0	0	0	0	0	0
6	3	9	0	0	1	4	2	3
7	1	0	0	0	0	0	0	0
7	2	0	0	0	0	0	0	0
7	3	9	0	0	1	5	2	4
8	1	2	1	0	0	4	3	4
8	2	5	0	1	0	4	3	4
8	3	0	0	0	0	0	0	0
9	1	0	0	0	0	0	0	0
9	2	5	0	1	0	6	1	1
9	3	0	0	0	0	0	0	0
10	1	2	1	0	0	5	2	2
10	2	5	0	1	0	5	2	2
10	3	9	0	0	1	5	2	2
11	1	0	0	0	0	0	0	0
11	2	4	0	1	0	1	5	2
11	3	0	0	0	0	0	0	0
12	1	2	1	0	0	3	1	4
12	2	4	0	1	0	3	1	4
12	3	9	0	0	1	3	1	4
13	1	1	1	0	0	2	2	1
13	2	5	0	1	0	2	2	1
13	3	7	0	0	1	2	2	1
14	1	0	0	0	0	0	0	0
14	2	4	0	1	0	1	4	2
14	3	0	0	0	0	0	0	0
15	1	0	0	0	0	0	0	0
15	2	5	0	1	0	4	4	1
15	3	7	0	0	1	4	4	1

PROBLEM 10 (continued)

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
16	1	0	0	0	0	0	0	0
16	2	4	0	1	0	3	5	5
16	3	0	0	0	0	0	0	0
17	1	0	0	0	0	0	0	0
17	2	4	0	1	0	3	2	4
17	3	0	0	0	0	0	0	0
18	1	2	1	0	0	1	2	3
18	2	6	0	1	0	1	2	3
18	3	0	0	0	0	0	0	0
19	1	3	1	0	0	5	2	5
19	2	5	0	1	0	5	2	5
19	3	7	0	0	1	5	2	5
20	1	3	1	0	0	5	2	4
20	2	6	0	1	0	5	2	4
20	3	0	0	0	0	0	0	0

PRECEDENCE RELATIONSHIPS

JOB 1 precedes: JOB 2 JOB 3 JOB 4 JOB 5 JOB 6
JOB 7
JOB 2 precedes: JOB 8
JOB 3 precedes: JOB 11
JOB 4 precedes: JOB 10 JOB 11
JOB 5 precedes: JOB 9 JOB 13
JOB 6 precedes: JOB 11 JOB 12
JOB 7 precedes: JOB 8
JOB 8 precedes: JOB 14 JOB 15
JOB 9 precedes: JOB 14
JOB 10 precedes: JOB 15 JOB 18
JOB 11 precedes: JOB 14 JOB 17
JOB 12 precedes: JOB 16
JOB 13 precedes: JOB 15 JOB 19
JOB 14 precedes: JOB 20
JOB 15 precedes: JOB 20
JOB 16 precedes: JOB 20
JOB 17 precedes: JOB 20
JOB 18 precedes: JOB 20
JOB 19 precedes: JOB 20
JOB 20 precedes: -

PROBLEM 11: 20 JOBS, SHAPE#2, MEDIUM COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	3	1	0	0	3	1	5
1	2	6	0	1	0	3	1	5
1	3	7	0	0	1	3	1	5
2	1	0	0	0	0	0	0	0
2	2	4	0	1	0	3	1	4
2	3	7	0	0	1	3	1	4
3	1	1	1	0	0	3	3	2
3	2	6	0	1	0	3	3	2
3	3	0	0	0	0	0	0	0
4	1	2	1	0	0	2	2	5
4	2	4	0	1	0	2	2	5
4	3	7	0	0	1	2	2	5
5	1	2	1	0	0	4	2	1
5	2	4	0	1	0	4	2	1
5	3	9	0	0	1	4	2	1
6	1	1	1	0	0	5	3	1
6	2	4	0	1	0	5	3	1
6	3	8	0	0	1	5	3	1
7	1	2	1	0	0	3	1	6
7	2	4	0	1	0	3	1	6
7	3	0	0	0	0	0	0	0
8	1	0	0	0	0	0	0	0
8	2	4	0	1	0	3	6	2
8	3	9	0	0	1	3	6	2
9	1	3	1	0	0	1	1	4
9	2	6	0	1	0	1	1	4
9	3	7	0	0	1	1	1	4
10	1	1	1	0	0	2	3	3
10	2	5	0	1	0	2	3	3
10	3	0	0	0	0	0	0	0
11	1	2	1	0	0	2	5	5
11	2	0	0	0	0	0	0	0
11	3	0	0	0	0	0	0	0
12	1	2	1	0	0	4	4	5
12	2	4	0	1	0	4	4	5
12	3	7	0	0	1	4	4	5
13	1	2	1	0	0	3	2	5
13	2	0	0	0	0	0	0	0
13	3	7	0	0	1	3	2	5
14	1	2	1	0	0	2	2	1
14	2	4	0	1	0	2	2	1
14	3	9	0	0	1	2	2	1
15	1	2	1	0	0	3	1	3
15	2	6	0	1	0	3	1	3
15	3	0	0	0	0	0	0	0

PROBLEM 11 (continued)

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
16	1	3	1	0	0	2	6	4
16	2	0	0	0	0	0	0	0
16	3	0	0	0	0	0	0	0
17	1	2	1	0	0	3	2	1
17	2	6	0	1	0	3	2	1
17	3	7	0	0	1	3	2	1
18	1	2	1	0	0	4	6	6
18	2	4	0	1	0	4	6	6
18	3	7	0	0	1	4	6	6
19	1	1	1	0	0	3	2	2
19	2	6	0	1	0	3	2	2
19	3	8	0	0	1	3	2	2
20	1	2	1	0	0	5	2	1
20	2	6	0	1	0	5	2	1
20	3	7	0	0	1	5	2	1

PRECEDENCE RELATIONSHIPS

JOB 1 precedes: JOB 2 JOB 3 JOB 4 JOB 5 JOB 6 JOB 7
JOB 2 precedes: JOB 10 JOB 12
JOB 3 precedes: JOB 9 JOB 11 JOB 12
JOB 4 precedes: JOB 8 JOB 10 JOB 12 JOB 13
JOB 5 precedes: JOB 11
JOB 6 precedes: JOB 10
JOB 7 precedes: JOB 8 JOB 9 JOB 10 JOB 12 JOB 13
JOB 8 precedes: JOB 16 JOB 18 JOB 19
JOB 9 precedes: JOB 15 JOB 16
JOB 10 precedes: JOB 14 JOB 15 JOB 17 JOB 19
JOB 11 precedes: JOB 16 JOB 17 JOB 18
JOB 12 precedes: JOB 14 JOB 17 JOB 19
JOB 13 precedes: JOB 15 JOB 17
JOB 14 precedes: JOB 20
JOB 15 precedes: JOB 20
JOB 16 precedes: JOB 20
JOB 17 precedes: JOB 20
JOB 18 precedes: JOB 20
JOB 19 precedes: JOB 20
JOB 20 precedes: -

PROBLEM 12: 20 JOBS, SHAPE#2, HIGH COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	0	0	0	0	0	0	0
1	2	4	0	1	0	2	5	3
1	3	7	0	0	1	2	5	3
2	1	2	1	0	0	3	5	2
2	2	5	0	1	0	3	5	2
2	3	0	0	0	0	0	0	0
3	1	3	1	0	0	1	2	2
3	2	0	0	0	0	0	0	0
3	3	0	0	0	0	0	0	0
4	1	3	1	0	0	3	1	1
4	2	0	0	0	0	0	0	0
4	3	7	0	0	1	3	1	1
5	1	3	1	0	0	3	3	5
5	2	5	0	1	0	3	3	5
5	3	0	0	0	0	0	0	0
6	1	3	1	0	0	5	6	3
6	2	4	0	1	0	5	6	3
6	3	8	0	0	1	5	6	3
7	1	2	1	0	0	1	4	3
7	2	6	0	1	0	1	4	3
7	3	0	0	0	0	0	0	0
8	1	3	1	0	0	4	3	6
8	2	5	0	1	0	4	3	6
8	3	0	0	0	0	0	0	0
9	1	2	1	0	0	6	3	3
9	2	0	0	0	0	0	0	0
9	3	9	0	0	1	6	3	3
10	1	2	1	0	0	1	4	6
10	2	5	0	1	0	1	4	6
10	3	8	0	0	1	1	4	6
11	1	2	1	0	0	4	1	3
11	2	6	0	1	0	4	1	3
11	3	7	0	0	1	4	1	3
12	1	3	1	0	0	2	4	2
12	2	4	0	1	0	2	4	2
12	3	8	0	0	1	2	4	2
13	1	3	1	0	0	2	1	3
13	2	0	0	0	0	0	0	0
13	3	0	0	0	0	0	0	0
14	1	2	1	0	0	1	4	2
14	2	5	0	1	0	1	4	2
14	3	7	0	0	1	1	4	2
15	1	1	1	0	0	4	1	2
15	2	0	0	0	0	0	0	0
15	3	0	0	0	0	0	0	0

PROBLEM 12 (continued)

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
16	1	3	1	0	0	5	5	2
16	2	4	0	1	0	5	5	2
16	3	0	0	0	0	0	0	0
17	1	0	0	0	0	0	0	0
17	2	5	0	1	0	3	4	2
17	3	0	0	0	0	0	0	0
18	1	0	0	0	0	0	0	0
18	2	6	0	1	0	1	4	2
18	3	8	0	0	1	1	4	2
19	1	1	1	0	0	5	3	2
19	2	0	0	0	0	0	0	0
19	3	0	0	0	0	0	0	0
20	1	2	1	0	0	2	3	3
20	2	5	0	1	0	2	3	3
20	3	0	0	0	0	0	0	0

PRECEDENCE RELATIONSHIPS

JOB 1 precedes: JOB 2 JOB 3 JOB 4 JOB 5 JOB 6 JOB 7
JOB 2 precedes: JOB 8 JOB 10 JOB 11
JOB 3 precedes: JOB 8 JOB 9 JOB 12 JOB 13
JOB 4 precedes: JOB 8 JOB 9 JOB 10 JOB 11 JOB 12
JOB 5 precedes: JOB 8 JOB 9 JOB 11 JOB 12 JOB 13
JOB 6 precedes: JOB 8 JOB 9 JOB 10 JOB 12 JOB 13
JOB 7 precedes: JOB 10 JOB 12 JOB 13
JOB 8 precedes: JOB 15 JOB 17 JOB 18 JOB 19
JOB 9 precedes: JOB 15 JOB 17 JOB 18
JOB 10 precedes: JOB 15 JOB 16 JOB 17 JOB 18 JOB 19
JOB 11 precedes: JOB 14 JOB 15 JOB 16 JOB 18 JOB 19
JOB 12 precedes: JOB 14 JOB 16 JOB 17 JOB 18 JOB 19
JOB 13 precedes: JOB 14 JOB 15 JOB 16 JOB 17 JOB 18 JOB 19
JOB 14 precedes: JOB 20
JOB 15 precedes: JOB 20
JOB 16 precedes: JOB 20
JOB 17 precedes: JOB 20
JOB 18 precedes: JOB 20
JOB 19 precedes: JOB 20
JOB 20 precedes: -

PROBLEM 13: 30 JOBS, SHAPE#1, LOW COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	3	1	0	0	3	2	2
1	2	5	0	1	0	3	2	2
1	3	0	0	0	0	0	0	0
2	1	2	1	0	0	3	1	1
2	2	6	0	1	0	3	1	1
2	3	9	0	0	1	3	1	1
3	1	3	1	0	0	2	3	1
3	2	5	0	1	0	2	3	1
3	3	7	0	0	1	2	3	1
4	1	2	1	0	0	4	1	2
4	2	4	0	1	0	4	1	2
4	3	8	0	0	1	4	1	2
5	1	3	1	0	0	2	2	6
5	2	0	0	0	0	0	0	0
5	3	0	0	0	0	0	0	0
6	1	2	1	0	0	2	6	4
6	2	0	0	0	0	0	0	0
6	3	8	0	0	1	2	6	4
7	1	1	1	0	0	3	6	3
7	2	0	0	0	0	0	0	0
7	3	0	0	0	0	0	0	0
8	1	2	1	0	0	2	1	3
8	2	0	0	0	0	0	0	0
8	3	0	0	0	0	0	0	0
9	1	1	1	0	0	3	2	1
9	2	0	0	0	0	0	0	0
9	3	7	0	0	1	3	2	1
10	1	2	1	0	0	3	5	4
10	2	6	0	1	0	3	5	4
10	3	0	0	0	0	0	0	0
11	1	3	1	0	0	4	6	1
11	2	0	0	0	0	0	0	0
11	3	0	0	0	0	0	0	0
12	1	0	0	0	0	0	0	0
12	2	0	0	0	0	0	0	0
12	3	8	0	0	1	2	3	4
13	1	3	1	0	0	2	3	3
13	2	4	0	1	0	2	3	3
13	3	9	0	0	1	2	3	3
14	1	1	1	0	0	2	3	2
14	2	6	0	1	0	2	3	2
14	3	9	0	0	1	2	3	2
15	1	2	1	0	0	1	6	1
15	2	4	0	1	0	1	6	1
15	3	0	0	0	0	0	0	0

PROBLEM 13 (continued)

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
16	1	0	0	0	0	0	0	0
16	2	0	0	0	0	0	0	0
16	3	7	0	0	1	2	1	1
17	1	1	1	0	0	3	2	2
17	2	4	0	1	0	3	2	2
17	3	0	0	0	0	0	0	0
18	1	1	1	0	0	4	1	4
18	2	4	0	1	0	4	1	4
18	3	7	0	0	1	4	1	4
19	1	2	1	0	0	5	1	3
19	2	4	0	1	0	5	1	3
19	3	7	0	0	1	5	1	3
20	1	1	1	0	0	4	3	4
20	2	4	0	1	0	4	3	4
20	3	8	0	0	1	4	3	4
21	1	0	0	0	0	0	0	0
21	2	5	0	1	0	1	1	2
21	3	7	0	0	1	1	1	2
22	1	3	1	0	0	2	4	1
22	2	0	0	0	0	0	0	0
22	3	7	0	0	1	2	4	1
23	1	1	1	0	0	3	3	4
23	2	0	0	0	0	0	0	0
23	3	0	0	0	0	0	0	0
24	1	3	1	0	0	2	4	4
24	2	6	0	1	0	2	4	4
24	3	8	0	0	1	2	4	4
25	1	3	1	0	0	1	1	1
25	2	4	0	1	0	1	1	1
25	3	7	0	0	1	1	1	1
26	1	3	1	0	0	5	1	3
26	2	5	0	1	0	5	1	3
26	3	9	0	0	1	5	1	3
27	1	1	1	0	0	4	4	4
27	2	4	0	1	0	4	4	4
27	3	0	0	0	0	0	0	0
28	1	3	1	0	0	4	3	3
28	2	6	0	1	0	4	3	3
28	3	9	0	0	1	4	3	3
29	1	3	1	0	0	1	4	1
29	2	0	0	0	0	0	0	0
29	3	7	0	0	1	1	4	1
30	1	2	1	0	0	2	1	3
30	2	6	0	1	0	2	1	3
30	3	8	0	0	1	2	1	3

PRECEDENCE RELATIONSHIPS

JOB 1	precedes:	JOB 2	JOB 3	JOB 4	JOB 5		
JOB 2	precedes:	JOB 8					
JOB 3	precedes:	JOB 6	JOB 10				
JOB 4	precedes:	JOB 6	JOB 7				
JOB 5	precedes:	JOB 6	JOB 7	JOB 9	JOB 11		
JOB 6	precedes:	JOB 13	JOB 17				
JOB 7	precedes:	JOB 17					
JOB 8	precedes:	JOB 14	JOB 15				
JOB 9	precedes:	JOB 12	JOB 14				
JOB 10	precedes:	JOB 13					
JOB 11	precedes:	JOB 16					
JOB 12	precedes:	JOB 21					
JOB 13	precedes:	JOB 20	JOB 21				
JOB 14	precedes:	JOB 20					
JOB 15	precedes:	JOB 18	JOB 19	JOB 20	JOB 22	JOB 23	
JOB 16	precedes:	JOB 19	JOB 20				
JOB 17	precedes:	JOB 21	JOB 22				
JOB 18	precedes:	JOB 25	JOB 27	JOB 29			
JOB 19	precedes:	JOB 25					
JOB 20	precedes:	JOB 25	JOB 26	JOB 27			
JOB 21	precedes:	JOB 24	JOB 25	JOB 26	JOB 27		
JOB 22	precedes:	JOB 24					
JOB 23	precedes:	JOB 27	JOB 28				
JOB 24	precedes:	JOB 30					
JOB 25	precedes:	JOB 30					
JOB 26	precedes:	JOB 30					
JOB 27	precedes:	JOB 30					
JOB 28	precedes:	JOB 30					
JOB 29	precedes:	JOB 30					
JOB 30	precedes:	-					

PROBLEM 14: 30 JOBS, SHAPE#1, MEDIUM COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION					
			TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	3	1	0	0	3	1	2
1	2	4	0	1	0	3	1	2
1	3	7	0	0	1	3	1	2
2	1	3	1	0	0	4	1	2
2	2	5	0	1	0	4	1	2
2	3	9	0	0	1	4	1	2
3	1	1	1	0	0	1	2	5
3	2	5	0	1	0	1	2	5
3	3	9	0	0	1	1	2	5
4	1	0	0	0	0	0	0	0
4	2	5	0	1	0	3	5	1
4	3	0	0	0	0	0	0	0
5	1	2	1	0	0	2	2	3
5	2	4	0	1	0	2	2	3
5	3	0	0	0	0	0	0	0
6	1	3	1	0	0	2	1	3
6	2	0	0	0	0	0	0	0
6	3	0	0	0	0	0	0	0
7	1	0	0	0	0	0	0	0
7	2	5	0	1	0	2	4	4
7	3	0	0	0	0	0	0	0
8	1	2	1	0	0	1	2	4
8	2	0	0	0	0	0	0	0
8	3	8	0	0	1	1	2	4
9	1	0	0	0	0	0	0	0
9	2	0	0	0	0	0	0	0
9	3	9	0	0	1	1	3	5
10	1	2	1	0	0	4	2	2
10	2	5	0	1	0	4	2	2
10	3	0	0	0	0	0	0	0
11	1	1	1	0	0	4	5	5
11	2	0	0	0	0	0	0	0
11	3	7	0	0	1	4	5	5
12	1	3	1	0	0	1	1	1
12	2	0	0	0	0	0	0	0
12	3	0	0	0	0	0	0	0
13	1	2	1	0	0	2	1	6
13	2	5	0	1	0	2	1	6
13	3	0	0	0	0	0	0	0
14	1	3	1	0	0	4	2	5
14	2	4	0	1	0	4	2	5
14	3	7	0	0	1	4	2	5
15	1	3	1	0	0	1	3	3
15	2	0	0	0	0	0	0	0
15	3	8	0	0	1	1	3	3

PROBLEM 14 (continued)

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
16	1	0	0	0	0	0	0	0
16	2	5	0	1	0	4	4	4
16	3	0	0	0	0	0	0	0
17	1	0	0	0	0	0	0	0
17	2	5	0	1	0	1	1	2
17	3	0	0	0	0	0	0	0
18	1	3	1	0	0	3	2	1
18	2	0	0	0	0	0	0	0
18	3	8	0	0	1	3	2	1
19	1	2	1	0	0	5	2	1
19	2	4	0	1	0	5	2	1
19	3	0	0	0	0	0	0	0
20	1	3	1	0	0	2	6	2
20	2	0	0	0	0	0	0	0
20	3	9	0	0	1	2	6	2
21	1	1	1	0	0	6	1	2
21	2	5	0	1	0	6	1	2
21	3	0	0	0	0	0	0	0
22	1	3	1	0	0	2	1	3
22	2	0	0	0	0	0	0	0
22	3	8	0	0	1	2	1	3
23	1	3	1	0	0	3	1	1
23	2	5	0	1	0	3	1	1
23	3	7	0	0	1	3	1	1
24	1	0	0	0	0	0	0	0
24	2	6	0	1	0	1	6	2
24	3	0	0	0	0	0	0	0
25	1	3	1	0	0	4	1	1
25	2	4	0	1	0	4	1	1
25	3	8	0	0	1	4	1	1
26	1	1	1	0	0	4	4	4
26	2	6	0	1	0	4	4	4
26	3	7	0	0	1	4	4	4
27	1	1	1	0	0	4	1	4
27	2	6	0	1	0	4	1	4
27	3	7	0	0	1	4	1	4
28	1	3	1	0	0	1	1	1
28	2	0	0	0	0	0	0	0
28	3	7	0	0	1	1	1	1
29	1	1	1	0	0	2	2	3
29	2	0	0	0	0	0	0	0
29	3	0	0	0	0	0	0	0
30	1	0	0	0	0	0	0	0
30	2	6	0	1	0	5	1	2
30	3	9	0	0	1	5	1	2

PRECEDENCE RELATIONSHIPS

JOB 1	precedes:	JOB 2	JOB 3	JOB 4	JOB 5	
JOB 2	precedes:	JOB 7	JOB 8	JOB 9	JOB 11	
JOB 3	precedes:	JOB 6	JOB 8	JOB 9	JOB 11	
JOB 4	precedes:	JOB 6	JOB 8	JOB 10		
JOB 5	precedes:	JOB 6	JOB 7	JOB 8	JOB 9	JOB 11
JOB 6	precedes:	JOB 12	JOB 16	JOB 17		
JOB 7	precedes:	JOB 13	JOB 14	JOB 15	JOB 16	JOB 17
JOB 8	precedes:	JOB 12	JOB 13	JOB 14	JOB 16	JOB 17
JOB 9	precedes:	JOB 16	JOB 17			
JOB 10	precedes:	JOB 13	JOB 17			
JOB 11	precedes:	JOB 12	JOB 13	JOB 14	JOB 17	
JOB 12	precedes:	JOB 18	JOB 20	JOB 22		
JOB 13	precedes:	JOB 20	JOB 21	JOB 22	JOB 23	
JOB 14	precedes:	JOB 18	JOB 19	JOB 21		
JOB 15	precedes:	JOB 19	JOB 20	JOB 21	JOB 22	JOB 23
JOB 16	precedes:	JOB 20	JOB 23			
JOB 17	precedes:	JOB 18	JOB 20	JOB 22		
JOB 18	precedes:	JOB 25	JOB 27			
JOB 19	precedes:	JOB 26	JOB 28			
JOB 20	precedes:	JOB 25	JOB 27			
JOB 21	precedes:	JOB 24	JOB 26	JOB 27	JOB 28	
JOB 22	precedes:	JOB 24	JOB 25	JOB 26	JOB 27	JOB 28
		JOB 29				
JOB 23	precedes:	JOB 27	JOB 28			
JOB 24	precedes:	JOB 30				
JOB 25	precedes:	JOB 30				
JOB 26	precedes:	JOB 30				
JOB 27	precedes:	JOB 30				
JOB 28	precedes:	JOB 30				
JOB 29	precedes:	JOB 30				
JOB 30	precedes:	-				

PROBLEM 15: 30 JOBS, SHAPE#1, HIGH COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	2	1	0	0	1	4	4
1	2	5	0	1	0	1	4	4
1	3	0	0	0	0	0	0	0
2	1	3	1	0	0	3	2	3
2	2	6	0	1	0	3	2	3
2	3	7	0	0	1	3	2	3
3	1	2	1	0	0	1	3	1
3	2	0	0	0	0	0	0	0
3	3	0	0	0	0	0	0	0
4	1	3	1	0	0	4	1	3
4	2	5	0	1	0	4	1	3
4	3	9	0	0	1	4	1	3
5	1	1	1	0	0	3	1	2
5	2	6	0	1	0	3	1	2
5	3	0	0	0	0	0	0	0
6	1	1	1	0	0	3	2	3
6	2	4	0	1	0	3	2	3
6	3	8	0	0	1	3	2	3
7	1	1	1	0	0	4	3	1
7	2	0	0	0	0	0	0	0
7	3	0	0	0	0	0	0	0
8	1	1	1	0	0	3	1	3
8	2	4	0	1	0	3	1	3
8	3	7	0	0	1	3	1	3
9	1	3	1	0	0	2	4	4
9	2	0	0	0	0	0	0	0
9	3	7	0	0	1	2	4	4
10	1	0	0	0	0	0	0	0
10	2	4	0	1	0	4	1	3
10	3	0	0	0	0	0	0	0
11	1	2	1	0	0	3	2	1
11	2	0	0	0	0	0	0	0
11	3	7	0	0	1	3	2	1
12	1	1	1	0	0	3	2	2
12	2	4	0	1	0	3	2	2
12	3	0	0	0	0	0	0	0
13	1	1	1	0	0	3	3	2
13	2	0	0	0	0	0	0	0
13	3	0	0	0	0	0	0	0
14	1	2	1	0	0	3	2	3
14	2	0	0	0	0	0	0	0
14	3	9	0	0	1	3	2	3
15	1	2	1	0	0	1	4	4
15	2	0	0	0	0	0	0	0
15	3	8	0	0	1	1	4	4

PROBLEM 15 (continued)

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
16	1	3	1	0	0	2	4	1
16	2	5	0	1	0	2	4	1
16	3	0	0	0	0	0	0	0
17	1	2	1	0	0	3	3	1
17	2	0	0	0	0	0	0	0
17	3	7	0	0	1	3	3	1
18	1	2	1	0	0	6	2	1
18	2	0	0	0	0	0	0	0
18	3	0	0	0	0	0	0	0
19	1	0	0	0	0	0	0	0
19	2	0	0	0	0	0	0	0
19	3	7	0	0	1	1	2	2
20	1	1	1	0	0	3	3	2
20	2	6	0	1	0	3	3	2
20	3	7	0	0	1	3	3	2
21	1	2	1	0	0	2	2	3
21	2	4	0	1	0	2	2	3
21	3	8	0	0	1	2	2	3
22	1	0	0	0	0	0	0	0
22	2	4	0	1	0	3	6	2
22	3	9	0	0	1	3	6	2
23	1	1	1	0	0	5	2	3
23	2	0	0	0	0	0	0	0
23	3	0	0	0	0	0	0	0
24	1	3	1	0	0	2	6	5
24	2	0	0	0	0	0	0	0
24	3	0	0	0	0	0	0	0
25	1	2	1	0	0	1	1	4
25	2	4	0	1	0	1	1	4
25	3	0	0	0	0	0	0	0
26	1	1	1	0	0	4	4	3
26	2	0	0	0	0	0	0	0
26	3	0	0	0	0	0	0	0
27	1	2	1	0	0	2	2	4
27	2	0	0	0	0	0	0	0
27	3	0	0	0	0	0	0	0
28	1	1	1	0	0	2	3	3
28	2	0	0	0	0	0	0	0
28	3	0	0	0	0	0	0	0
29	1	0	0	0	0	0	0	0
29	2	4	0	1	0	3	3	1
29	3	0	0	0	0	0	0	0
30	1	0	0	0	0	0	0	0
30	2	4	0	1	0	3	1	1
30	3	8	0	0	1	3	1	1

PRECEDENCE RELATIONSHIPS

JOB 1	precedes:	JOB 2 JOB 3 JOB 4 JOB 5
JOB 2	precedes:	JOB 6 JOB 9 JOB 10 JOB 11
JOB 3	precedes:	JOB 7 JOB 8 JOB 9 JOB 10
JOB 4	precedes:	JOB 9 JOB 11
JOB 5	precedes:	JOB 6 JOB 8 JOB 9 JOB 10
JOB 6	precedes:	JOB 12 JOB 13 JOB 14 JOB 16 JOB 17
JOB 7	precedes:	JOB 12 JOB 15 JOB 16 JOB 17
JOB 8	precedes:	JOB 12 JOB 13 JOB 14 JOB 15 JOB 16 JOB 17
JOB 9	precedes:	JOB 12 JOB 13 JOB 14 JOB 15 JOB 17
JOB 10	precedes:	JOB 12 JOB 13 JOB 17
JOB 11	precedes:	JOB 12 JOB 13 JOB 15 JOB 16 JOB 17
JOB 12	precedes:	JOB 18 JOB 19 JOB 20 JOB 21 JOB 22 JOB 23
JOB 13	precedes:	JOB 18 JOB 19 JOB 20 JOB 22 JOB 23
JOB 14	precedes:	JOB 18 JOB 19 JOB 21 JOB 22
JOB 15	precedes:	JOB 18 JOB 19 JOB 20 JOB 21 JOB 22 JOB 23
JOB 16	precedes:	JOB 18 JOB 19 JOB 20 JOB 21 JOB 22
JOB 17	precedes:	JOB 19 JOB 20 JOB 21 JOB 22 JOB 23
JOB 18	precedes:	JOB 24 JOB 27 JOB 28
JOB 19	precedes:	JOB 25 JOB 26 JOB 27 JOB 28 JOB 29
JOB 20	precedes:	JOB 25 JOB 26 JOB 27 JOB 28 JOB 29
JOB 21	precedes:	JOB 24 JOB 27 JOB 28
JOB 22	precedes:	JOB 25 JOB 26 JOB 27 JOB 29
JOB 23	precedes:	JOB 25 JOB 26 JOB 27 JOB 28 JOB 29
JOB 24	precedes:	JOB 30
JOB 25	precedes:	JOB 30
JOB 26	precedes:	JOB 30
JOB 27	precedes:	JOB 30
JOB 28	precedes:	JOB 30
JOB 29	precedes:	JOB 30
JOB 30	precedes:	-

PROBLEM 16: 30 JOBS, SHAPE#2, LOW COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	2	1	0	0	4	4	2
1	2	0	0	0	0	0	0	0
1	3	7	0	0	1	4	4	2
2	1	3	1	0	0	4	5	2
2	2	0	0	0	0	0	0	0
2	3	0	0	0	0	0	0	0
3	1	0	0	0	0	0	0	0
3	2	6	0	1	0	4	1	3
3	3	8	0	0	1	4	1	3
4	1	3	1	0	0	4	2	3
4	2	0	0	0	0	0	0	0
4	3	8	0	0	1	4	2	3
5	1	0	0	0	0	0	0	0
5	2	6	0	1	0	2	3	4
5	3	0	0	0	0	0	0	0
6	1	1	1	0	0	2	3	5
6	2	5	0	1	0	2	3	5
6	3	7	0	0	1	2	3	5
7	1	2	1	0	0	2	4	4
7	2	0	0	0	0	0	0	0
7	3	0	0	0	0	0	0	0
8	1	1	1	0	0	5	2	4
8	2	0	0	0	0	0	0	0
8	3	8	0	0	1	5	2	4
9	1	0	0	0	0	0	0	0
9	2	5	0	1	0	2	4	3
9	3	0	0	0	0	0	0	0
10	1	2	1	0	0	1	4	5
10	2	0	0	0	0	0	0	0
10	3	0	0	0	0	0	0	0
11	1	0	0	0	0	0	0	0
11	2	4	0	1	0	3	3	3
11	3	8	0	0	1	3	3	3
12	1	3	1	0	0	3	5	3
12	2	4	0	1	0	3	5	3
12	3	7	0	0	1	3	5	3
13	1	2	1	0	0	6	2	3
13	2	6	0	1	0	6	2	3
13	3	7	0	0	1	6	2	3
14	1	1	1	0	0	5	3	2
14	2	6	0	1	0	5	3	2
14	3	7	0	0	1	5	3	2
15	1	3	1	0	0	3	4	6
15	2	0	0	0	0	0	0	0
15	3	0	0	0	0	0	0	0

PROBLEM 16 (continued)

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
16	1	2	1	0	0	1	2	4
16	2	4	0	1	0	1	2	4
16	3	7	0	0	1	1	2	4
17	1	0	0	0	0	0	0	0
17	2	0	0	0	0	0	0	0
17	3	8	0	0	1	1	3	6
18	1	0	0	0	0	0	0	0
18	2	4	0	1	0	2	2	1
18	3	0	0	0	0	0	0	0
19	1	1	1	0	0	6	4	5
19	2	4	0	1	0	6	4	5
19	3	7	0	0	1	6	4	5
20	1	3	1	0	0	6	3	3
20	2	4	0	1	0	6	3	3
20	3	7	0	0	1	6	3	3
21	1	0	0	0	0	0	0	0
21	2	5	0	1	0	1	4	3
21	3	0	0	0	0	0	0	0
22	1	0	0	0	0	0	0	0
22	2	6	0	1	0	1	2	1
22	3	0	0	0	0	0	0	0
23	1	1	1	0	0	2	1	2
23	2	4	0	1	0	2	1	2
23	3	0	0	0	0	0	0	0
24	1	1	1	0	0	2	1	6
24	2	4	0	1	0	2	1	6
24	3	0	0	0	0	0	0	0
25	1	0	0	0	0	0	0	0
25	2	6	0	1	0	4	4	1
25	3	0	0	0	0	0	0	0
26	1	1	1	0	0	1	1	2
26	2	4	0	1	0	1	1	2
26	3	8	0	0	1	1	1	2
27	1	1	1	0	0	1	1	1
27	2	4	0	1	0	1	1	1
27	3	0	0	0	0	0	0	0
28	1	1	1	0	0	2	1	2
28	2	5	0	1	0	2	1	2
28	3	8	0	0	1	2	1	2
29	1	1	1	0	0	3	3	3
29	2	5	0	1	0	3	3	3
29	3	9	0	0	1	3	3	3
30	1	0	0	0	0	0	0	0
30	2	4	0	1	0	6	2	3
30	3	8	0	0	1	6	2	3

PRECEDENCE RELATIONSHIPS

JOB 1	precedes:	JOB 2 JOB 3 JOB 4 JOB 5 JOB 6 JOB 7 JOB 8
JOB 2	precedes:	JOB 10 JOB 13
JOB 3	precedes:	JOB 9 JOB 10 JOB 14 JOB 15
JOB 4	precedes:	JOB 14
JOB 5	precedes:	JOB 9
JOB 6	precedes:	JOB 11 JOB 12 JOB 13
JOB 7	precedes:	JOB 13
JOB 8	precedes:	JOB 13
JOB 9	precedes:	JOB 16 JOB 17 JOB 19 JOB 21
JOB 10	precedes:	JOB 17 JOB 19
JOB 11	precedes:	JOB 17
JOB 12	precedes:	JOB 22
JOB 13	precedes:	JOB 16 JOB 20
JOB 14	precedes:	JOB 16 JOB 17 JOB 18 JOB 19 JOB 20
JOB 15	precedes:	JOB 17
JOB 16	precedes:	JOB 24
JOB 17	precedes:	JOB 25 JOB 29
JOB 18	precedes:	JOB 26 JOB 27
JOB 19	precedes:	JOB 29
JOB 20	precedes:	JOB 23 JOB 28
JOB 21	precedes:	JOB 25 JOB 26 JOB 27 JOB 29
JOB 22	precedes:	JOB 23
JOB 23	precedes:	JOB 30
JOB 24	precedes:	JOB 30
JOB 25	precedes:	JOB 30
JOB 26	precedes:	JOB 30
JOB 27	precedes:	JOB 30
JOB 28	precedes:	JOB 30
JOB 29	precedes:	JOB 30
JOB 30	precedes:	-

PROBLEM 17: 30 JOBS, SHAPE#2, MEDIUM COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	3	1	0	0	2	1	1
1	2	0	0	0	0	0	0	0
1	3	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0
2	2	0	0	0	0	0	0	0
2	3	9	0	0	1	5	3	1
3	1	1	1	0	0	4	2	2
3	2	0	0	0	0	0	0	0
3	3	7	0	0	1	4	2	2
4	1	1	1	0	0	1	2	5
4	2	0	0	0	0	0	0	0
4	3	0	0	0	0	0	0	0
5	1	2	1	0	0	3	2	2
5	2	0	0	0	0	0	0	0
5	3	0	0	0	0	0	0	0
6	1	0	0	0	0	0	0	0
6	2	5	0	1	0	1	5	3
6	3	8	0	0	1	1	5	3
7	1	3	1	0	0	3	4	1
7	2	5	0	1	0	3	4	1
7	3	8	0	0	1	3	4	1
8	1	2	1	0	0	1	1	4
8	2	6	0	1	0	1	1	4
8	3	8	0	0	1	1	1	4
9	1	0	0	0	0	0	0	0
9	2	4	0	1	0	1	1	3
9	3	8	0	0	1	1	1	3
10	1	0	0	0	0	0	0	0
10	2	6	0	1	0	2	1	3
10	3	7	0	0	1	2	1	3
11	1	1	1	0	0	4	1	3
11	2	4	0	1	0	4	1	3
11	3	0	0	0	0	0	0	0
12	1	3	1	0	0	5	4	4
12	2	4	0	1	0	5	4	4
12	3	0	0	0	0	0	0	0
13	1	0	0	0	0	0	0	0
13	2	6	0	1	0	4	4	1
13	3	0	0	0	0	0	0	0
14	1	0	0	0	0	0	0	0
14	2	4	0	1	0	2	1	3
14	3	0	0	0	0	0	0	0
15	1	2	1	0	0	2	4	4
15	2	0	0	0	0	0	0	0
15	3	9	0	0	1	2	4	4

PROBLEM 17 (continued)

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
16	1	0	0	0	0	0	0	0
16	2	0	0	0	0	0	0	0
16	3	9	0	0	1	4	4	4
17	1	3	1	0	0	2	6	6
17	2	4	0	1	0	2	6	6
17	3	9	0	0	1	2	6	6
18	1	2	1	0	0	1	5	2
18	2	4	0	1	0	1	5	2
18	3	0	0	0	0	0	0	0
19	1	2	1	0	0	3	5	2
19	2	4	0	1	0	3	5	2
19	3	0	0	0	0	0	0	0
20	1	2	1	0	0	2	3	3
20	2	6	0	1	0	2	3	3
20	3	0	0	0	0	0	0	0
21	1	3	1	0	0	4	2	1
21	2	0	0	0	0	0	0	0
21	3	7	0	0	1	4	2	1
22	1	3	1	0	0	4	6	3
22	2	4	0	1	0	4	6	3
22	3	7	0	0	1	4	6	3
23	1	0	0	0	0	0	0	0
23	2	4	0	1	0	5	2	5
23	3	0	0	0	0	0	0	0
24	1	3	1	0	0	2	3	1
24	2	6	0	1	0	2	3	1
24	3	9	0	0	1	2	3	1
25	1	0	0	0	0	0	0	0
25	2	6	0	1	0	1	3	4
25	3	8	0	0	1	1	3	4
26	1	0	0	0	0	0	0	0
26	2	5	0	1	0	2	3	2
26	3	0	0	0	0	0	0	0
27	1	1	1	0	0	1	2	3
27	2	6	0	1	0	1	2	3
27	3	0	0	0	0	0	0	0
28	1	1	1	0	0	6	1	1
28	2	5	0	1	0	6	1	1
28	3	0	0	0	0	0	0	0
29	1	2	1	0	0	4	2	1
29	2	4	0	1	0	4	2	1
29	3	0	0	0	0	0	0	0
30	1	2	1	0	0	1	5	2
30	2	4	0	1	0	1	5	2
30	3	7	0	0	1	1	5	2

PRECEDENCE RELATIONSHIPS

JOB 1	precedes:	JOB 2	JOB 3	JOB 4	JOB 5	JOB 6
		JOB 7	JOB 8			
JOB 2	precedes:	JOB 9	JOB 13	JOB 14		
JOB 3	precedes:	JOB 9	JOB 11	JOB 12	JOB 14	
JOB 4	precedes:	JOB 9	JOB 11	JOB 12	JOB 14	JOB 15
JOB 5	precedes:	JOB 12	JOB 15			
JOB 6	precedes:	JOB 13	JOB 15			
JOB 7	precedes:	JOB 9	JOB 10	JOB 15		
JOB 8	precedes:	JOB 9	JOB 10	JOB 11	JOB 13	JOB 14
JOB 9	precedes:	JOB 16	JOB 21	JOB 22		
JOB 10	precedes:	JOB 19	JOB 20			
JOB 11	precedes:	JOB 20	JOB 22			
JOB 12	precedes:	JOB 18	JOB 19	JOB 20	JOB 21	JOB 22
JOB 13	precedes:	JOB 16	JOB 20	JOB 22		
JOB 14	precedes:	JOB 16	JOB 20	JOB 21	JOB 22	
JOB 15	precedes:	JOB 16	JOB 17	JOB 20	JOB 21	JOB 22
JOB 16	precedes:	JOB 24	JOB 29			
JOB 17	precedes:	JOB 23	JOB 29			
JOB 18	precedes:	JOB 25	JOB 27	JOB 28		
JOB 19	precedes:	JOB 24	JOB 25	JOB 27		
JOB 20	precedes:	JOB 23	JOB 24	JOB 26	JOB 27	JOB 28
		JOB 29				
JOB 21	precedes:	JOB 25	JOB 26			
JOB 22	precedes:	JOB 25				
JOB 23	precedes:	JOB 30				
JOB 24	precedes:	JOB 30				
JOB 25	precedes:	JOB 30				
JOB 26	precedes:	JOB 30				
JOB 27	precedes:	JOB 30				
JOB 28	precedes:	JOB 30				
JOB 29	precedes:	JOB 30				
JOB 30	precedes:	-				

PROBLEM 18: 30 JOBS, SHAPE#2, HIGH COMPLEXITY

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
1	1	0	0	0	0	0	0	0
1	2	4	0	1	0	3	3	2
1	3	8	0	0	1	3	3	2
2	1	0	0	0	0	0	0	0
2	2	5	0	1	0	2	1	3
2	3	0	0	0	0	0	0	0
3	1	3	1	0	0	5	4	2
3	2	5	0	1	0	5	4	2
3	3	0	0	0	0	0	0	0
4	1	3	1	0	0	1	5	5
4	2	0	0	0	0	0	0	0
4	3	0	0	0	0	0	0	0
5	1	3	1	0	0	3	2	1
5	2	0	0	0	0	0	0	0
5	3	9	0	0	1	3	2	1
6	1	2	1	0	0	1	4	2
6	2	0	0	0	0	0	0	0
6	3	0	0	0	0	0	0	0
7	1	1	1	0	0	1	1	4
7	2	4	0	1	0	1	1	4
7	3	8	0	0	1	1	1	4
8	1	3	1	0	0	2	1	3
8	2	0	0	0	0	0	0	0
8	3	8	0	0	1	2	1	3
9	1	2	1	0	0	4	3	3
9	2	6	0	1	0	4	3	3
9	3	9	0	0	1	4	3	3
10	1	1	1	0	0	3	1	1
10	2	5	0	1	0	3	1	1
10	3	7	0	0	1	3	1	1
11	1	3	1	0	0	3	1	5
11	2	5	0	1	0	3	1	5
11	3	8	0	0	1	3	1	5
12	1	2	1	0	0	1	3	6
12	2	0	0	0	0	0	0	0
12	3	0	0	0	0	0	0	0
13	1	3	1	0	0	3	2	5
13	2	4	0	1	0	3	2	5
13	3	0	0	0	0	0	0	0
14	1	2	1	0	0	6	2	3
14	2	6	0	1	0	6	2	3
14	3	0	0	0	0	0	0	0
15	1	2	1	0	0	2	1	2
15	2	0	0	0	0	0	0	0
15	3	0	0	0	0	0	0	0

PROBLEM 18 (continued)

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5	TYPE6
16	1	3	1	0	0	5	4	1
16	2	6	0	1	0	5	4	1
16	3	9	0	0	1	5	4	1
17	1	1	1	0	0	3	5	1
17	2	6	0	1	0	3	5	1
17	3	0	0	0	0	0	0	0
18	1	0	0	0	0	0	0	0
18	2	6	0	1	0	3	3	1
18	3	0	0	0	0	0	0	0
19	1	2	1	0	0	5	1	4
19	2	5	0	1	0	5	1	4
19	3	8	0	0	1	5	1	4
20	1	2	1	0	0	3	5	2
20	2	4	0	1	0	3	5	2
20	3	7	0	0	1	3	5	2
21	1	2	1	0	0	4	1	1
21	2	0	0	0	0	0	0	0
21	3	0	0	0	0	0	0	0
22	1	2	1	0	0	4	4	2
22	2	4	0	1	0	4	4	2
22	3	0	0	0	0	0	0	0
23	1	1	1	0	0	2	2	5
23	2	4	0	1	0	2	2	5
23	3	0	0	0	0	0	0	0
24	1	3	1	0	0	4	3	4
24	2	0	0	0	0	0	0	0
24	3	0	0	0	0	0	0	0
25	1	1	1	0	0	5	2	3
25	2	0	0	0	0	0	0	0
25	3	0	0	0	0	0	0	0
26	1	2	1	0	0	6	4	5
26	2	0	0	0	0	0	0	0
26	3	0	0	0	0	0	0	0
27	1	0	0	0	0	0	0	0
27	2	4	0	1	0	2	3	1
27	3	0	0	0	0	0	0	0
28	1	3	1	0	0	2	5	3
28	2	4	0	1	0	2	5	3
28	3	0	0	0	0	0	0	0
29	1	2	1	0	0	2	4	5
29	2	0	0	0	0	0	0	0
29	3	8	0	0	1	2	4	5
30	1	2	1	0	0	3	1	4
30	2	6	0	1	0	3	1	4
30	3	8	0	0	1	3	1	4

PRECEDENCE RELATIONSHIPS

```
JOB 1  precedes:  JOB 2  JOB 3  JOB 4  JOB 5  JOB 6
                  JOB 7  JOB 8
JOB 2  precedes:  JOB 10 JOB 12 JOB 14 JOB 15
JOB 3  precedes:  JOB 9  JOB 10 JOB 11 JOB 12 JOB 13
                  JOB 14 JOB 15
JOB 4  precedes:  JOB 9  JOB 10 JOB 11 JOB 12 JOB 14
                  JOB 15
JOB 5  precedes:  JOB 11 JOB 12 JOB 13 JOB 14 JOB 15
JOB 6  precedes:  JOB 9  JOB 11 JOB 12 JOB 13 JOB 14
JOB 7  precedes:  JOB 10 JOB 11 JOB 12 JOB 13 JOB 15
JOB 8  precedes:  JOB 9  JOB 10 JOB 12 JOB 13 JOB 14
                  JOB 15
JOB 9  precedes:  JOB 16 JOB 17 JOB 18 JOB 19 JOB 21
                  JOB 22
JOB 10 precedes:  JOB 17 JOB 19 JOB 21 JOB 22
JOB 11 precedes:  JOB 16 JOB 18 JOB 20 JOB 22
JOB 12 precedes:  JOB 16 JOB 17 JOB 18 JOB 20 JOB 21
JOB 13 precedes:  JOB 17 JOB 18 JOB 19 JOB 21 JOB 22
JOB 14 precedes:  JOB 16 JOB 17 JOB 19 JOB 21
JOB 15 precedes:  JOB 16 JOB 17 JOB 18 JOB 19 JOB 20
                  JOB 21 JOB 22
JOB 16 precedes:  JOB 23 JOB 24 JOB 25 JOB 26 JOB 27
                  JOB 29
JOB 17 precedes:  JOB 23 JOB 24 JOB 25 JOB 27 JOB 29
JOB 18 precedes:  JOB 23 JOB 24 JOB 25 JOB 26 JOB 27
                  JOB 28 JOB 29
JOB 19 precedes:  JOB 24 JOB 25
JOB 20 precedes:  JOB 23 JOB 24 JOB 25 JOB 29
JOB 21 precedes:  JOB 23 JOB 25 JOB 26 JOB 27 JOB 29
JOB 22 precedes:  JOB 24 JOB 25 JOB 26 JOB 28 JOB 29
JOB 23 precedes:  JOB 30
JOB 24 precedes:  JOB 30
JOB 25 precedes:  JOB 30
JOB 26 precedes:  JOB 30
JOB 27 precedes:  JOB 30
JOB 28 precedes:  JOB 30
JOB 29 precedes:  JOB 30
JOB 30 precedes:   -
```

SAMPLE PROBLEM[1]

JOB	MODE	DURATION	RENEWABLE RESOURCE CONSUMPTION TYPE1	TYPE2	TYPE3	TYPE4	TYPE5
1	1	2	1	0	2	1	135
1	2	3	0	1	2	1	65
2	1	1	1	0	3	2	160
2	2	3	0	1	3	2	90
3	1	3	1	0	1	4	170
3	2	4	0	1	1	4	100
4	1	5	1	0	1	3	155
4	2	7	0	1	1	3	85
5	1	4	1	0	2	2	150
5	2	6	0	1	2	2	80
6	1	1	1	0	3	4	190
6	2	4	0	1	3	4	120
7	1	0	0	0	0	0	0

PRECEDENCE RELATIONSHIPS

JOB 1 precedes: JOB 4 JOB 5
JOB 2 precedes: JOB 6
JOB 3 precedes: JOB 7
JOB 4 precedes: JOB 6
JOB 5 precedes: JOB 7
JOB 6 precedes: JOB 7
JOB 7 precedes: -

NOTE: [1] refer to the sample problem from Talbot's paper (1982).

APPENDIX 3

PROBLEM SETS FOR TESTING THE ALGORITHMS

DATA D1

PROBLEM	RENEWABLE RESOURCE LIMITATION OF TYPE (1,2,3,4,5,6)	CRITICAL PATH EARLY FINISH (a)	OPTIMAL PROJECT COMPLETION TIME
1	1,2,1,8,9,9	16	21
2	1,2,1,5,8,6	16	26
3	1,2,1,7,6,6	23	29
4	1,2,1,6,9,6	11	21
5	1,2,1,6,6,6	13	21
6	1,2,1,7,7,7	16	18
7	1,2,1,17,17,17	35	36
8	2,1,1,8,8,8	27	31
9	3,3,3,9,9,9	26	30
10	3,3,3,14,13,14	20	21
11	3,3,3,15,15,15	14	16
12	2,2,2,14,13,14	19	23
13	4,4,4,12,12,12	25	26
14	2,2,2,12,12,12	35	36
15	2,2,2,11,11,11	26	29
16	2,2,2,14,13,14	31	31
17	2,3,1,13,15,13	32	32
18	4,4,4,15,15,14	23	25

(a) The critical path early finish is obtained when no resource restrictions are imposed.

DATA D2

PROBLEM	RENEWABLE RESOURCE LIMITATION OF TYPE (1,2,3,4,5,6)	UNIT COST ASSOCIATED WITH RENEWABLE RESOURCE TYPE (1,2,3,4,5,6)	TOTAL COST LIMITATION (a)
1	1,2,1,9,9,9	80,20,10,5,5,5	2605
2	1,1,2,6,6,6	200,20,10,5,5,5	3605
3	1,2,1,7,7,7	160,20,10,5,5,5	3660
4	1,2,1,7,7,7	80,20,10,5,5,5	2620
5	1,2,1,6,6,6	80,20,10,5,5,5	2840
6	1,2,2,9,7,9	80,20,10,5,5,5	2900
7	2,2,2,13,13,13	30,20,10,5,5,5	4530
8	2,1,1,10,10,10	70,20,10,5,5,5	5195
9	3,3,3,12,12,12	50,20,10,5,5,5	4050
10	3,3,3,15,15,15	100,20,10,5,5,5	6245
11	4,4,4,15,15,15	100,50,10,5,5,5	5885
12	5,5,5,15,15,15	100,20,10,5,5,5	6270
13	4,4,4,15,15,15	100,20,10,5,5,5	8730
14	3,3,3,12,12,12	50,20,10,5,5,5	7000
15	3,3,3,12,12,12	100,20,10,5,5,5	7290
16	2,2,2,14,14,14	100,20,10,5,5,5	8405
17	3,3,3,15,15,15	100,20,10,5,5,5	9550
18	5,5,5,20,20,20	100,20,10,5,5,5	8625
SAMPLE[b]	1,2,6,8,300	100,30,10,15,0	2020

D2 (continued)

PROBLEM	CRITICAL PATH EARLY FINISH	OPTIMAL PROJECT COMPLETION TIME
1	16	18
2	16	37
3	23	34
4	11	20
5	13	21
6	16	24
7	35	35
8	27	28
9	26	27
10	20	20
11	14	15
12	19	22
13	25	25
14	35	35
15	26	28
16	31	31
17	32	32
18	23	26
SAMPLE[b]	8	11

(a) Cost is considered as nonrenewable resource.

(b) SAMPLE is the sample problem taken from Talbot's paper (1982). In the sample probelm, only five renewable resource types are included. Furthermore, the resource type 5, cost, is considered as a doubly-constrained resource.

DATA D3

PROBLEM	UNIT COST ASSOCIATED WITH RENEWABLE RESOURCE TYPE (1,2,3,4,5,6)	DESIRED FINISH TIME	OPTIMAL SOLUTION (F) or KNOWN HEURISTIC SOLUTION (H)
1	200,20,10,5,5,5	25	3605 (F)
2	200,20,10,5,5,5	30	3835 (F)
3	160,20,10,5,5,5	32	3750 (F)
4	80,20,10,5,5,5	24	2615 (F)
5	80,20,10,5,5,5	27	2755 (F)
6	80,20,10,5,5,5	29	2885 (F)
7	30,20,10,5,5,5	40	4695 (H)
8	70,20,10,5,5,5	34	5000 (F)
9	100,20,10,5,5,5	35	5120 (F)
10	100,20,10,5,5,5	24	6020 (H)
11	100,50,10,5,5,5	20	5755 (F)
12	100,20,10,5,5,5	26	5990 (F)
13	100,20,10,5,5,5	30	8030 (H)
14	50,20,10,5,5,5	40	6830 (F)
15	100,20,10,5,5,5	34	7090 (H)
16	160,20,10,5,5,5	35	9500 (H)
17	100,20,10,5,5,5	40	8610 (H)
18	100,20,10,5,5,5	30	8595 (H)
SAMPLE	100,30,10,15,0	10	2200 (F)

NOTE: The renewable resource limitation for each resource type is the same as D1.

DATA D4

PROBLEM	UNIT COST ASSOCIATED WITH RENEWABLE RESOURCE TYPE (1,2,3,4,5,6)	DESIRED FINISH TIME	OPTIMAL SOLUTION (F) or KNOWN HEURISTIC SOLUTION (H)
1	200,20,10,5,5,5	25	3605 (F)
2	200,20,10,5,5,5	30	3835 (F)
3	160,20,10,5,5,5	32	3750 (F)
4	80,20,10,5,5,5	24	2615 (F)
5	80,20,10,5,5,5	27	2755 (F)
6	80,20,10,5,5,5	29	2885 (F)
8	70,20,10,5,5,5	34	5000 (F)
9	100,20,10,5,5,5	35	5120 (F)
11	100,50,10,5,5,5	20	5755 (F)
SAMPLE	100,30,10,15,0	10	2200 (F)

NOTE: (1) The renewable resource limitation for each resource type is the same as D1.

(2) The tested problems in D4 are selected from the tested problems in D3 based on their computational time and final solutions.

DATA D5

PROBLEM	DESIRED FINISH TIME	OPTIMAL SOLUTION (F) or KNOWN HEURISTIC SOLUTION (H)
1	25	299 (F)
2	30	304 (F)
3	32	315 (F)
4	24	249 (F)
5	27	176 (F)
6	29	310 (F)
7	40	1241 (H)
8	34	325 (H)
9	35	611 (H)
10	24	374 (H)
11	20	418 (H)
12	26	501 (H)
13	30	616 (H)
14	40	732 (H)
15	34	593 (H)
16	35	607 (H)
17	40	555 (H)
18	30	736 (H)

NOTE: The renewable resource limitation for each resource type is the same as D1.